Chapter 5

Discovery

The greatest value of a picture is when it forces us to notice what we never expected to see.
—John W. Tukey, *Exploratory Data Analysis*

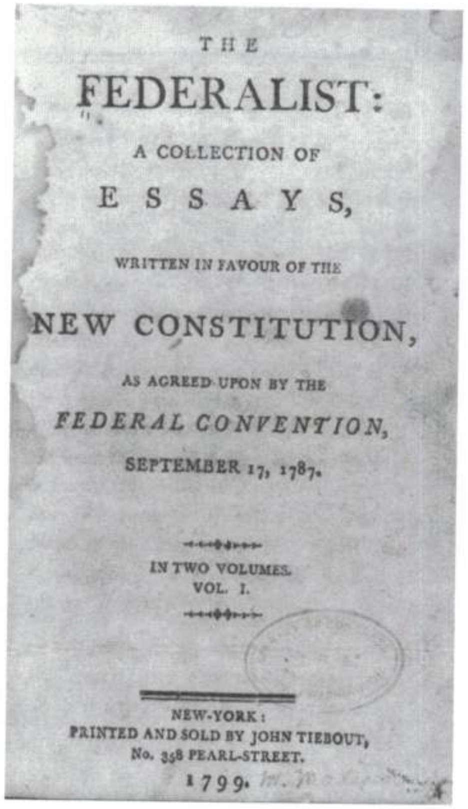
Over the last couple of decades, the variety as well as volume of data analyzed in quantitative social science research has dramatically increased. In this chapter, we introduce three types of data that were not analyzed in previous chapters: textual, network, and spatial data. We conduct *exploratory data analysis* to inductively learn about the underlying patterns and structure of these data. We saw an example of such analysis applied to the degree of political polarization in chapter 3. In this chapter, we first analyze textual data to discover topics and predict authorship of documents based on the frequency of word usage. Our application is the disputed authorship of *The Federalist Papers*. Second, we analyze network data, which record the relationships among units. As examples, we will explore the marriage network in Renaissance Florence and social media data from Twitter. Finally, we visualize spatial data and examine changes in patterns across time and space. Our examples are the cholera outbreak in the 19th century and the expansion of Walmart retail stores in the 21st century.

5.1 Textual Data

The widespread use of the Internet has led to an astronomical amount of digitized textual data accumulating every second through email, websites, and social media outlets. The analysis of blog sites and social media posts can give new insights into human behavior and opinions. At the same time, large-scale efforts to digitize published articles, books, and government documents have been underway, providing exciting opportunities to revisit previously studied questions, by analyzing new data.

5.1.1 THE DISPUTED AUTHORSHIP OF *THE FEDERALIST PAPERS*

While new opportunities for text analysis have grown in recent years, we begin by revisiting one of the earliest examples of text analysis in the statistics literature. We analyze the text of *The Federalist*, more commonly known as *The Federalist*



THE
FEDERALIST:
A COLLECTION OF
ESSAYS,
WRITTEN IN FAVOUR OF THE
NEW CONSTITUTION,
AS AGREED UPON BY THE
FEDERAL CONVENTION,
SEPTEMBER 17, 1787.

IN TWO VOLUMES.
VOL. I.

NEW-YORK:
PRINTED AND SOLD BY JOHN TIEBOUT,
No. 358 PEARL-STREET.
1799.

Figure 5.1. The Title Page of *The Federalist*, Vol. 1. Source: Library of Congress.

Papers.[1] *The Federalist*, whose title page is displayed in figure 5.1, consists of 85 essays attributed to Alexander Hamilton, John Jay, and James Madison from 1787 to 1788 in order to encourage people in New York to ratify the newly drafted US Constitution. Because both Hamilton and Madison helped draft the Constitution, scholars regard *The Federalist Papers* as a primary document reflecting the intentions of the authors of the Constitution.

The Federalist Papers were originally published in various New York state newspapers under the pseudonym of "Publius." For this reason, the authorship of each paper has been the subject of scholarly research. According to the Library of Congress,[2] experts believe that Hamilton wrote 51 essays while Madison authored 15.[3] In addition, Hamilton and Madison jointly authored 3 papers whereas John Jay wrote 5.[4] The remaining 11 essays were written by either Hamilton or Madison, though

[1] This section is in part based on F. Mosteller and D.L. Wallace (1963) "Inference in an authorship problem." *Journal of the American Statistical Association*, vol. 58, no. 302, pp. 275–309.

[2] See the website `https://www.congress.gov/resources/display/content/The+Federalist+Papers#TheFederalistPapers-1`.

[3] *The Federalist Papers* known to be written by Hamilton: nos. 1, 6–9, 11–13, 15–17, 21–36, 59–61, and 65–85. Papers known to be written by Madison: nos. 10, 14, 37–48, and 58.

[4] *The Federalist Papers* known to be jointly written by Hamilton and Madison: nos. 18–20. *The Federalist Papers* known to be written by John Jay: nos. 2–5 and 64.

Table 5.1. *The Federalist Papers* Data.

```
AFTER an unequivocal experience of the inefficiency
of the subsisting federal government, you are called
upon to deliberate on a new Constitution for the
United States of America.
                        ⋮
This shall accordingly constitute the subject of my next
address.
```

Note: The data consists of the raw text of each of 85 essays in *The Federalist Papers*. The first and last sentences of *The Federalist Paper* no. 1 appear here as an example.

scholars dispute which one.[5] Below, we analyze the text of *The Federalist Papers* to predict their authorship.

The text of the 85 essays is scraped from the Library of Congress website and stored as `fpXX.txt`, where `XX` represents the essay number ranging from `01` to `85`. *Scraping* refers to an automated method of data collection from websites using a computer program. Each data file contains the textual data of its corresponding essay. See table 5.1, which displays the first and last sentences of *The Federalist Paper* no. 1 as an example.

Before analyzing the data, we need to preprocess it. The **tm** package provides a number of useful *natural language processing* functionalities in R. One functionality eliminates unnecessary white space between words. Another, called stemming, strips away prefixes and suffixes to produce stem words so that different forms of the same word can be recognized. For example, the stem form of "government" is "govern." Note that the stemming functionality in the **tm** package requires another package called **SnowballC**. Be sure to install these packages by utilizing the `install.packages()` function or clicking the `Install` icon under the `Packages` tab in the bottom-right window of RStudio (see section 1.3.7 for more detailed instructions). The installation of a package needs only to occur once. However, in order to use a package, you must load it once in each new R session using the `library()` function. Load multiple packages simultaneously by separating them with commas.

```
## load two required libraries
library(tm, SnowballC)
```

We begin by loading the text *corpus*, or collection of texts, into R using the `Corpus()` function. The `DirSource()` function specifies the directory and pattern of corpus file names. The `directory` argument indicates the files' location, in this case the working directory's subdirectory called `federalist`, a folder you must create

[5] *The Federalist Papers* with disputed authorship are nos. 49, 50–57, 62, and 63.

Table 5.2. Commonly Used Functions to Preprocess Raw Texts.

Function	*Description*
`tolower()`	transform to lower case
`stripWhitespace()`	remove white space
`removePunctuation()`	remove punctuation
`removeNumbers()`	remove numbers
`removeWords()`	remove specified words
`stemDocument()`	stem the words in a document for specified language

before running the code. The `pattern` argument identifies a pattern contained in the names of all data files, in this case `fp` (`fp01.txt`, `fp10.txt`, etc.).

```
## load the raw corpus
corpus.raw <- VCorpus(DirSource(directory = "federalist", pattern = "fp"))
corpus.raw
## <<VCorpus>>
## Metadata:  corpus specific: 0, document level (indexed): 0
## Content:  documents: 85
```

We now preprocess our corpus. We use the `tm_map()` function, which enables various natural language processing operations on corpora. The first argument of this function is the name of a corpus, while the second argument is a function that transforms text. Table 5.2 summarizes these functions. We first turn all letters to lower case by using the `tolower()` function. Since `tolower()` is a function in the R **base** package rather than in the **tm** package, it must pass through the wrapper function called `content_transformer()` (as of version 0.6–1).[6] Next, we eliminate unnecessary white space with the `stripWhitespace()` function, remove punctuation with the `removePunctuation()` function, and remove numbers with the `removeNumbers()` function.

```
## make lower case
corpus.prep <- tm_map(corpus.raw, content_transformer(tolower))
## remove white space
corpus.prep <- tm_map(corpus.prep, stripWhitespace)
## remove punctuation
corpus.prep <- tm_map(corpus.prep, removePunctuation)
## remove numbers
corpus.prep <- tm_map(corpus.prep, removeNumbers)
```

[6] Note that older versions of the **tm** package do not require the use of the `content_transformer()` function.

Next, to remove the most commonly used words such as a and the, we first use the stopwords() function to obtain a list of stop words for the input language. The beginning of the English list appears below.

```
head(stopwords("english"))
## [1] "i"       "me"      "my"      "myself" "we"      "our"
```

We will then pass this list through the removeWords() function. Finally, we stem each word.

```
## remove stop words
corpus <- tm_map(corpus.prep, removeWords, stopwords("english"))
## finally stem remaining words
corpus <- tm_map(corpus, stemDocument)
```

We can extract a specific essay by using double square brackets [[and]] with an integer indicating the element to be extracted (see section 3.7.2 for more details about the use of double square brackets). In addition, the content() function prints out the actual text of the selected document.

```
## the output is truncated here to save space
content(corpus[[10]]) # essay no. 10
##   [1] "among  numer advantag promis   wellconstruct union none"
##   [2] " deserv    accur develop   tendenc  break "
##   [3] " control  violenc  faction  friend  popular govern never"
...
```

Compare this preprocessed document with the corresponding section of the original text, which is displayed here.

```
AMONG the numerous advantages promised by a well-constructed
   Union, none
    deserves to be more accurately developed than its tendency
   to break and
    control the violence of faction. The friend of popular
   governments never
```

We observe from the above text that all preprocessing was done as specified in our prior code. That is, all letters were transformed to lower case, punctuation marks such as hyphens and commas were taken out, stop words and white space were removed, and words were stemmed to be reduced to their stem word (e.g., transform numerous to numer and promised to promis).

5.1.2 DOCUMENT-TERM MATRIX

One quick way to explore textual data is to simply count occurrences of each word or term. The number of times a particular word appears in a given document is called *term frequency* (*tf*). The tf statistic can be summarized in a *document-term matrix*, which is a rectangular array with rows representing documents and columns representing unique terms. The (i, j) element of this matrix gives the counts of the jth term (column) in the ith document (row). We can also flip rows and columns and convert a document-term matrix to a *term-document matrix* where rows and columns represent terms and documents, respectively. A document-term matrix can be created by the `DocumentTermMatrix()` function in R (similarly, the `TermDocumentMatrix()` function creates a term-document matrix).

```
dtm <- DocumentTermMatrix(corpus)
dtm

## <<DocumentTermMatrix (documents: 85, terms: 4849)>>
## Non-/sparse entries: 44917/367248
## Sparsity           : 89%
## Maximal term length: 18
## Weighting          : term frequency (tf)
```

Because the output of the `DocumentTermMatrix()` function is a special matrix, R prints the document-term matrix's summary rather than the document-term matrix itself. The summary contains the number of documents as well as the number of terms. In addition, the number of nonsparse or nonzero entries and the number of sparse entries in the document-term matrix are provided. `Sparsity` refers to the proportion of zero entries in the document-term matrix. As is the case in this example, a document-term matrix is typically *sparse*. That is, the vast majority of its entries are zero because most terms appear in only a small number of documents. In the case of *The Federalist Papers*, 89% of the elements of the document-term matrix are 0. Finally, the summary output provides the maximal term length and quantity by which the entries of this matrix are weighted. In the current example, each entry represents the tf statistic.

To take a closer look at the actual entries of this matrix, we use the `inspect()` function, which displays detailed information on a corpus or term-document matrix. We can subset these matrix objects just like we subset a data frame object using square brackets `[,]`. As an example, the following syntax inspects the first 5 rows and first 8 columns of the document-term matrix.

```
inspect(dtm[1:5, 1:8])

## <<DocumentTermMatrix (documents: 5, terms: 8)>>
## Non-/sparse entries: 4/36
## Sparsity           : 90%
## Maximal term length: 7
## Weighting          : term frequency (tf)
##
```

```
##           Terms
## Docs      abandon abat abb abet abhorr abil abject abl
##   fp01.txt      0    0   0    0      0    0      0   1
##   fp02.txt      0    0   0    0      0    1      0   0
##   fp03.txt      0    0   0    0      0    0      0   2
##   fp04.txt      0    0   0    0      0    0      0   1
##   fp05.txt      0    0   0    0      0    0      0   0
```

Alternatively, we can coerce this object into a standard `matrix` object using the `as.matrix()` function, and print it directly.

```
dtm.mat <- as.matrix(dtm)
```

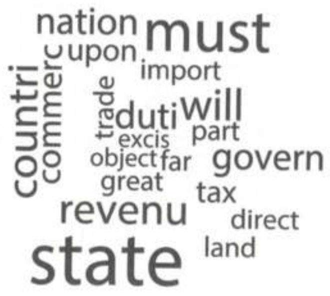
5.1.3 TOPIC DISCOVERY

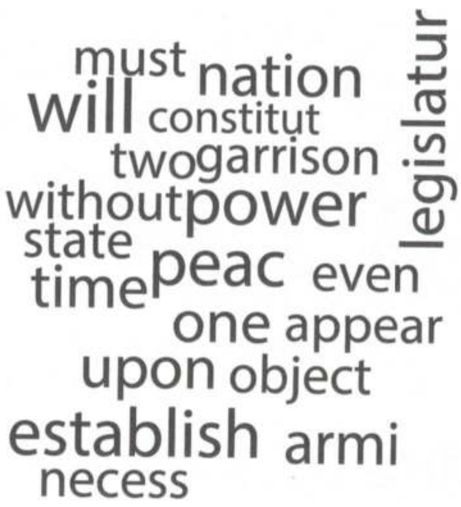
We begin by visualizing and analyzing the document-term matrix created above. Our analysis of word frequency critically relies on the commonly used *bag-of-words* assumption, which ignores the grammar and ordering of words. This means that our analysis is unlikely to detect subtle meanings of texts. The distribution of *term frequency* (tf) should, however, allow us to infer *topics* discussed in the documents. A common way to visualize this distribution is a *word cloud* where more frequently used words appear in a larger font. The `wordcloud()` function in the **wordcloud** package creates a word cloud, which may serve as a useful visualization tool because the document-term matrix often contains too many columns to visually inspect.

Like clustering, covered in section 3.7, topic discovery is an example of *unsupervised learning* because we lack access to true information about topic assignment. That is, we do not know, a priori, what topics exist in the corpus and characterize each document. We wish to discover topics by analyzing the distribution of term frequency within a given document and across documents. In contrast, in *supervised learning*, researchers use a sample with an observed outcome variable to learn about the relationship between the outcome and predictors. For example, we may have human coders read some documents and assign topics. We can then use this information to predict the topics of other documents that have not been read. Clearly, the lack of information about outcome variables makes unsupervised learning problems more challenging than supervised problems.

We begin by visualizing *The Federalist Papers* nos. 12 and 24 with word clouds in order to infer their topics. Both papers are known to be authored by Alexander Hamilton. In the **wordcloud** package, which we must install, the `wordcloud()` function takes two main arguments. The first argument takes a vector of words while the second argument takes the frequencies of those words. To avoid clutter, we limit the maximum number of words to be plotted by setting `max.words` to `20`.

```
library(wordcloud)
wordcloud(colnames(dtm.mat), dtm.mat[12, ], max.words = 20)  # essay no. 12
wordcloud(colnames(dtm.mat), dtm.mat[24, ], max.words = 20)  # essay no. 24
```





The comparison of the two word clouds shows that the left-hand plot for paper no. 12 contains words related to economy such as `revenu` (the root form of `revenue`), `commerc` (`commerce`), `trade`, `tax`, `land`, and so on. In contrast, the right-hand plot for paper no. 24 contains more words about security including `power`, `peac` (the root form of `peace`), `garrison`, and `armi` (`army`). Recall that the `stemDocument()` function stems documents. We now can use the `stemCompletion()` function to recover the full version of a stemmed word. The function's first argument takes the stem word or words, while the second argument takes candidate full words. Our candidate full words here come from the unstemmed corpus, `corpus.prep`.

```
stemCompletion(c("revenu", "commerc", "peac", "army"), corpus.prep)
##      revenu    commerc        peac        army
##   "revenue" "commerce"    "peace"      "army"
```

These discovered topics are indeed consistent with the actual content of the papers. Paper no. 12 is entitled, "The utility of the Union in respect to revenue" and discusses the economic benefits of the 13 colonies forming one nation. In contrast, the title of no. 24 is "The powers necessary to the common defense further considered" and discusses the creation of a national army as well as the relationship between legislative power and federal forces.

In the above analysis, we visualized the distribution of term frequency within each document. However, a certain term's high frequency within a document means little if that term often appears across the documents of the corpus. To address this issue, we should downweight the terms that occur frequently across documents. This can be done by computing the statistic called *term frequency–inverse document frequency*, or *tf–idf* in short. The tf–idf statistic is another measure of the importance of each term in a given document. For a given document d and term w, we define tf–idf(w, d) as

$$\text{tf–idf}(w, d) = \text{tf}(w, d) \times \text{idf}(w). \tag{5.1}$$

In the above equation, tf(w, d) represents *term frequency* or the number of occurrences of term w in document d. In some cases, we convert tf(w, d) to a log scale when it takes a positive value. Note that tf(w, d) equals 0 when term w never occurs in document d.

The other factor in equation (5.1), idf(w), is the *inverse document frequency*, which is typically defined as

$$\text{idf}(w) = \log\left(\frac{N}{\text{df}(w)}\right).$$

In this equation, N is the total number of documents and df(w) is the *document frequency* or the number of documents that contain term w. Dividing by df(w) implies that idf(w) takes a smaller value when term w is used more frequently across documents. As a consequence, common terms across documents receive less weight in tf–idf.

We can compute the tf–idf measure using the `weightTfIdf()` function, which takes as its input the document-term matrix output from the `DocumentTermMatrix()` function. Note that the `weightTfIdf()` function has an argument `normalize`, for which the default value is `FALSE`. If this argument is set to `TRUE`, then term frequency tf(w, d) will be divided by the total number of terms in document d.

```
dtm.tfidf <- weightTfIdf(dtm) # tf-idf calculation
```

Below, we list the 10 most important terms for *The Federalist Papers* nos. 12 and 24 using the tf–idf measure. The `sort()` function helpfully identifies the terms with the largest tf–idf values. We sort a vector in decreasing (increasing) order by specifying the `decreasing` argument as `TRUE` (`FALSE`). Since the class of `dtm.tfidf` is still `DocumentTermMatrix`, we need to convert it to a matrix before applying the `sort()` function.

```
dtm.tfidf.mat <- as.matrix(dtm.tfidf)  # convert to matrix
## 10 most important words for paper no. 12
head(sort(dtm.tfidf.mat[12, ], decreasing = TRUE), n = 10)
##     revenu contraband     patrol      excis      coast
## 0.01905877 0.01886965 0.01886965 0.01876560 0.01592559
##      trade        per        tax       cent     gallon
## 0.01473504 0.01420342 0.01295466 0.01257977 0.01257977
## 10 most important words for paper no. 24
head(sort(dtm.tfidf.mat[24, ], decreasing = TRUE), n = 10)
##   garrison   dockyard settlement      spain       armi
## 0.02965511 0.01962294 0.01962294 0.01649040 0.01544256
##   frontier    arsenal    western       post     nearer
## 0.01482756 0.01308196 0.01306664 0.01236780 0.01166730
```

The results clearly show that the most important terms for *The Federalist Paper* no. 12 concern the economy whereas those for paper no. 24 relate to security policies, though such word association is done by the researcher.

The analysis of documents based on term frequency relies on the **bag-of-words** assumption that ignores the order of words. To measure the relative importance of a term in a document, we can compute the **term frequency–inverse document frequency** (tf–idf), which represents the relative frequency of the term inversely weighted by the number of documents in which the term appears (document frequency).

Finally, we consider an alternative approach to topic discovery, by identifying clusters of similar essays, based on the tf–idf measure. We focus on the essays written by Hamilton. Following section 3.7, we apply the *k*-means algorithm to this weighted document-term matrix. After some experimentation, we choose the number of clusters to be 4. While arbitrary, this choice produces clusters that seem reasonable. We check the number of iterations to convergence to make sure that it does not exceed the default maximum value `10`.

```
k <- 4  # number of clusters
## subset the Federalist papers written by Hamilton
hamilton <- c(1, 6:9, 11:13, 15:17, 21:36, 59:61, 65:85)
dtm.tfidf.hamilton <- dtm.tfidf.mat[hamilton, ]
## run k-means
km.out <- kmeans(dtm.tfidf.hamilton, centers = k)
km.out$iter # check the convergence; number of iterations may vary
## [1] 2
```

We next summarize the results by printing out the 10 most important terms at the centroid of each of the resulting clusters. We also show which essays of *The Federalist Papers* belong to each cluster. Since we must perform the same operation for each cluster, we use a loop (see section 4.1.1).

```
## label each centroid with the corresponding term
colnames(km.out$centers) <- colnames(dtm.tfidf.hamilton)
for (i in 1:k) { # loop for each cluster
    cat("CLUSTER", i, "\n")
    cat("Top 10 words:\n") # 10 most important terms at the centroid
    print(head(sort(km.out$centers[i, ], decreasing = TRUE), n = 10))
    cat("\n")
    cat("Federalist Papers classified:\n") # extract essays classified
    print(rownames(dtm.tfidf.hamilton)[km.out$cluster == i])
    cat("\n")
}
## CLUSTER 1
## Top 10 words:
##     vacanc     recess      claus      senat    session
## 0.06953047 0.04437713 0.04082617 0.03408008 0.03313305
##       fill    appoint     presid      expir    unfound
## 0.03101140 0.02211662 0.01852025 0.01738262 0.01684465
##
## Federalist Papers classified:
## [1] "fp67.txt"
##
## CLUSTER 2
## Top 10 words:
##        armi        upon     militia      revenu        land
## 0.004557667 0.003878185 0.003680496 0.003523467 0.003410589
##    militari         war confederaci       taxat      esourc
## 0.003378875 0.003035943 0.003021217 0.002835844 0.002699460
##
## Federalist Papers classified:
##  [1] "fp01.txt" "fp06.txt" "fp07.txt" "fp08.txt" "fp09.txt"
##  [6] "fp11.txt" "fp12.txt" "fp13.txt" "fp15.txt" "fp16.txt"
## [11] "fp17.txt" "fp21.txt" "fp22.txt" "fp23.txt" "fp24.txt"
## [16] "fp25.txt" "fp26.txt" "fp27.txt" "fp28.txt" "fp29.txt"
## [21] "fp30.txt" "fp31.txt" "fp34.txt" "fp35.txt" "fp36.txt"
## [26] "fp60.txt" "fp80.txt" "fp85.txt"
##
## CLUSTER 3
## Top 10 words:
##       senat      presid       claus       offic     impeach
## 0.008267389 0.007114606 0.005340963 0.005134467 0.005124293
##       nomin    governor     appoint        upon     magistr
## 0.004568173 0.004490385 0.003965382 0.003748606 0.003667998
##
## Federalist Papers classified:
##  [1] "fp32.txt" "fp33.txt" "fp59.txt" "fp61.txt" "fp65.txt"
##  [6] "fp66.txt" "fp68.txt" "fp69.txt" "fp70.txt" "fp71.txt"
## [11] "fp72.txt" "fp73.txt" "fp74.txt" "fp75.txt" "fp76.txt"
## [16] "fp77.txt" "fp78.txt" "fp79.txt" "fp84.txt"
##
## CLUSTER 4
## Top 10 words:
##      court       juri      appel  jurisdict     suprem
## 0.05119100 0.03715999 0.01948060 0.01865612 0.01474737
##     tribun      trial     cogniz   inferior     appeal
## 0.01448872 0.01383180 0.01343695 0.01155172 0.01139125
##
## Federalist Papers classified:
## [1] "fp81.txt" "fp82.txt" "fp83.txt"
```

Examining the 10 most important terms at the centroid of each cluster suggests that cluster 2 relates to war and taxation, as indicated by terms like `armi`, `taxat`, and `war`, while cluster 1 covers only one document. Cluster 3 addresses institutional design and cluster 4 appears to be concerned with judicial systems. Comparing these topics with the actual contents of *The Federalist Papers* shows a decent degree of validity for the results of the k-means clustering algorithm.

We have been using *The Federalist Papers* to illustrate how text analyses can reveal topics. Of course, since we can easily read all of *The Federalist Papers*, the automated text analysis may not be necessary in this case. However, similar and more advanced techniques can be applied to a much larger corpus that humans would struggle to read in full over a short amount of time. In such situations, automated text analysis can play an essential role in helping researchers extract meaningful information from textual data.

5.1.4 AUTHORSHIP PREDICTION

As mentioned earlier, the authorship of some of *The Federalist Papers* is unknown. We will use the 66 essays attributed to either Hamilton or Madison to predict the authorship of the 11 disputed papers. Since each *Federalist* paper deals with a different topic, we focus on the usage of adjectives, adverbs, prepositions and conjunctions. In particular, we analyze the frequency of the following 10 words: `although`, `always`, `commonly`, `consequently`, `considerable`, `enough`, `there`, `upon`, `while`, `whilst`. We select these words based on the analysis presented in the academic paper that inspired this section (see footnote 1). As a result, we must use the unstemmed corpus, `corpus.prep`. We first compute the term frequency (per 1000 words) separately for each term and document and then subset the resulting term-frequency matrix to contain only these words.

```
## document-term matrix converted to matrix for manipulation
dtm1 <- as.matrix(DocumentTermMatrix(corpus.prep))
tfm <- dtm1 / rowSums(dtm1) * 1000 # term frequency per 1000 words
```

```
## words of interest
words <- c("although", "always", "commonly", "consequently",
           "considerable", "enough", "there", "upon", "while", "whilst")
## select only these words
tfm <- tfm[, words]
```

We then calculate the average term frequency separately for Hamilton and Madison across each author's entire body of documents.

```
## essays written by Madison: "hamilton" defined earlier
madison <- c(10, 14, 37:48, 58)
## average among Hamilton/Madison essays
tfm.ave <- rbind(colSums(tfm[hamilton, ]) / length(hamilton),
                 colSums(tfm[madison, ]) / length(madison))
tfm.ave
##        although    always  commonly consequently
## [1,] 0.01756975 0.7527744 0.2630876   0.02600857
## [2,] 0.27058809 0.2006710 0.0000000   0.44878468
##      considerable    enough    there      upon     while
## [1,]    0.5435127 0.3955031 4.417750 4.3986828 0.3700484
## [2,]    0.1601669 0.0000000 1.113252 0.2000269 0.0000000
##           whilst
## [1,] 0.007055719
## [2,] 0.380113114
```

The results suggest that Hamilton prefers to use terms such as `there` and `upon`, which Madison seldom uses, preferring instead to use `consequently` and `whilst`. We will use the frequency of these 4 words as the predictors of a linear regression model, where the outcome variable is the authorship of an essay. We first fit this linear regression model to the 66 essays whose authorship is known to estimate the coefficients. The resulting fitted model can then be used to predict the unknown authorship of the 11 essays based on the 4 words' frequencies. For the linear regression model, we first create the outcome variable by coding essays authored by Hamilton as `1` and those written by Madison as `-1`. We then construct a data frame object, which contains this authorship variable as well as the term-frequency matrix `tfm` for all essays whose authorship is known.

```
author <- rep(NA, nrow(dtm1)) # a vector with missing values
author[hamilton] <- 1  # 1 if Hamilton
author[madison] <- -1  # -1 if Madison
## data frame for regression
author.data <- data.frame(author = author[c(hamilton,madison)],
                          tfm[c(hamilton, madison), ])
```

To predict the authorship, we use the term frequency of the 4 words selected based on our preliminary analysis, i.e., `upon`, `there`, `consequently`, and `whilst`. The data frame object we created above contains the term frequency of the 10 words including these 4. We estimate the coefficients using the 66 essays with known authorship.

```
hm.fit <- lm(author ~ upon + there + consequently + whilst,
             data = author.data)
hm.fit

##
## Call:
## lm(formula = author ~ upon + there + consequently + whilst, data = author.data)
##
## Coefficients:
##  (Intercept)          upon         there  consequently
##     -0.26288       0.16678       0.09494      -0.44012
##       whilst
##     -0.65875
```

The results are consistent with the preliminary analysis we conducted above. The estimated coefficients for `upon` and `there` are positive while those for `consequently` and `whilst` are negative, implying that the first two words are associated with Hamilton whereas the latter pair are associated with Madison. Interestingly, the estimated coefficient for `whilst` has the largest magnitude. Holding the term frequency of the other 3 words constant, one additional use of `whilst` (per 1000 words) in an essay decreases the predicted authorship score by 0.66. To put this number into perspective, we compute the standard deviation of fitted values using the `fitted()` and `sd()` functions.

```
hm.fitted <- fitted(hm.fit) # fitted values
sd(hm.fitted)

## [1] 0.7180769
```

We find that the magnitude of this coefficient is large and close to 1 standard deviation of fitted values. That is, one additional use of `whilst` (per 1000 words) accounts for approximately 1 standard deviation of variation in our predicted value for the authorship score.

5.1.5 CROSS VALIDATION

How well is this model fitting the data? We classify each essay using its fitted value and compute the *classification error*. To do this, we compute the proportion of positive fitted values among the essays authored by Hamilton. Similarly, we compute the proportion of negative fitted values among those written by Madison. The results represent the classification success rate (see section 4.1.3).

```
## proportion of correctly classified essays by Hamilton
mean(hm.fitted[author.data$author == 1] > 0)

## [1] 1

## proportion of correctly classified essays by Madison
mean(hm.fitted[author.data$author == -1] < 0)

## [1] 1
```

The results show that the model perfectly classifies the authorship of these essays. Like the coefficient of determination introduced in chapter 4, however, this measure of prediction accuracy is based on *in-sample prediction*. That is, the same data we used to fit the model are again used for assessing the prediction accuracy. This is not necessarily a good idea because we can *overfit* a model to the data at hand. Overfitting occurs when a model captures idiosyncratic features of a specific sample while muddling up systematic patterns that exist across different samples.

Let us instead consider *out-of-sample prediction*. The idea is that we use new observations to assess the predictive performance of a model. In chapter 4, we performed out-of-sample prediction by forecasting election results using preelection polls. Similarly, here, we employ a procedure called *leave-one-out cross validation*. Specifically, we set aside one observation and predict its outcome variable value after fitting the model to the remaining observations. We repeat this procedure for each observation in the sample and compute the classification error. Cross validation enables us to assess the accuracy of model prediction without relying on in-sample prediction.

Cross validation is a methodology to assess the accuracy of model prediction without relying on in-sample prediction, which often leads to overfitting. Suppose that we have a sample of n observations. Then, the leave-one-out cross-validation procedure repeats the following steps for each observation $i = 1, \ldots, n$:

1. Take out the ith observation and set it aside.
2. Fit the model using the remaining $n - 1$ observations.
3. Using the fitted model, predict the outcome for the ith observation and compute the prediction error.

Finally, compute the average prediction error across n observations as a measure of prediction accuracy.

In R, we can cross validate using a *loop*, where each iteration fits the model to the data after excluding one observation, then predicts that observation's outcome variable value. A convenient way of setting aside the ith observation is to use the minus sign, i.e., `-i`, to remove a certain row of the data frame. As we saw in section 4.3.4, the `predict()` function can compute the predicted value $\widehat{Y}$. In this function, the `newdata` argument should specify a data frame whose only row is the observation of interest.

```r
n <- nrow(author.data)
hm.classify <- rep(NA, n) # a container vector with missing values
for (i in 1:n) {
    ## fit the model to the data after removing the ith observation
    sub.fit <- lm(author ~ upon + there + consequently + whilst,
                  data = author.data[-i, ]) # exclude ith row
    ## predict the authorship for the ith observation
    hm.classify[i] <- predict(sub.fit, newdata = author.data[i, ])
}
```

The results below show that even when the cross validation procedure is used, the model continues to perfectly classify the authorship of each essay.

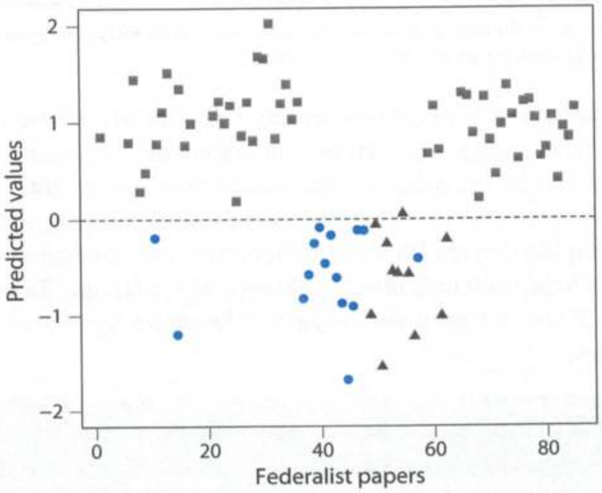
```r
## proportion of correctly classified essays by Hamilton
mean(hm.classify[author.data$author == 1] > 0)
## [1] 1
## proportion of correctly classified essays by Madison
mean(hm.classify[author.data$author == -1] < 0)
## [1] 1
```

Finally, we use this fitted model to predict the unknown authorship of the 11 essays. When using `predict()` for prediction, don't forget to coerce the term-frequency matrix into a data frame through the `as.data.frame()` function. Note that this function differs from the `data.frame()` function, which creates a data frame.

```r
disputed <- c(49, 50:57, 62, 63) # 11 essays with disputed authorship
tf.disputed <- as.data.frame(tfm[disputed, ])
## prediction of disputed authorship
pred <- predict(hm.fit, newdata = tf.disputed)
pred # predicted values
##    fp49.txt    fp50.txt    fp51.txt    fp52.txt    fp53.txt
## -0.99831799 -0.06759254 -1.53243206 -0.26288400 -0.54584900
##    fp54.txt    fp55.txt    fp56.txt    fp57.txt    fp62.txt
## -0.56566555  0.04376632 -0.57115610 -1.22289415 -1.00675456
##    fp63.txt
## -0.21939646
```

For ease of presentation, we plot the predicted values using different colors. Red squares signify essays known to be written by Hamilton, while blue circles indicate those by Madison. Black triangles represent papers with disputed authorship. Points above (below) the dashed horizontal line, indicating zero, correspond to essays classified as written by Hamilton (Madison).

```
## fitted values for essays authored by Hamilton; red squares
plot(hamilton, hm.fitted[author.data$author == 1], pch = 15,
     xlim = c(1, 85), ylim  = c(-2, 2), col = "red",
     xlab = "Federalist Papers", ylab = "Predicted values")
abline(h = 0, lty = "dashed")
## essays authored by Madison; blue circles
points(madison, hm.fitted[author.data$author == -1],
      pch = 16, col = "blue")
## disputed authorship; black triangles
points(disputed, pred, pch = 17)
```



The plot above uses gray squares instead of red squares for the essays authored by Hamilton. See page C4 for the full-color version. As our plot shows, the model predicts that Madison wrote all of the 11 essays except one. That one was barely classified as written by Hamilton, having a predicted value near zero.

5.2 Network Data

Next, we consider *network data*, which describes relationships among units rather than units in isolation. Examples include friendship networks among people, citation networks among academic articles, and trade and alliance networks among countries. Analysis of network data differs from the data analyses we have covered so far in that the unit of analysis is a relationship.

5.2.1 MARRIAGE NETWORK IN RENAISSANCE FLORENCE

We introduce the basic concepts and methods for network data by analyzing a well-known data set about the marriage network in Renaissance Florence.[7] The CSV data

[7] This section is in part based on John F. Padgett and Christopher K. Ansell (1993) "Robust action and the rise of the Medici, 1400–1434." *American Journal of Sociology*, vol. 98, no. 6, pp. 1259–1319.

Table 5.3. Florence Marriage Network Data.

FAMILY	ACCIAIUOL	ALBIZZI	⋯	LAMBERTES	MEDICI	⋯	STROZZI	TORNABUON
ACCIAIUOL	0	0	⋯	0	1	⋯	0	0
ALBIZZI	0	0	⋯	0	1	⋯	0	0
⋮			⋮			⋮		
LAMBERTES	0	0	⋯	0	0	⋯	0	0
MEDICI	1	1	⋯	0	0	⋯	0	1
⋮			⋮			⋮		
STROZZI	0	0	⋯	0	0	⋯	0	0
TORNABUON	0	0	⋯	0	1	⋯	0	0

Note: The data are in the form of an adjacency matrix where each entry represents whether a family in its row has a marriage relationship with another family in its column.

file, `florentine.csv`, contains an *adjacency matrix* whose entries represent the existence of relationships between two units (one unit represented by the row and the other represented by the column). Specifically, there are 16 elite Florentine families in the data, leading to a 16×16 adjacency matrix. If the (i, j) entry of this adjacency matrix is 1, then it implies that the ith and jth Florentine families had a marriage relationship. In contrast, a value of 0 indicates the absence of a marriage. Table 5.3 displays part of this data set. Below, we print out the part of the adjacency matrix corresponding to the first 5 families.

```
## the first column "FAMILY" of the CSV file represents row names
florence <- read.csv("florentine.csv", row.names = "FAMILY")
florence <- as.matrix(florence) # coerce into a matrix
## print out the adjacency (sub)matrix for the first 5 families
florence[1:5, 1:5]
##           ACCIAIUOL ALBIZZI BARBADORI BISCHERI CASTELLAN
## ACCIAIUOL         0       0         0        0         0
## ALBIZZI           0       0         0        0         0
## BARBADORI         0       0         0        0         1
## BISCHERI          0       0         0        0         0
## CASTELLAN         0       0         1        0         0
```

The submatrix shows that there was only one marriage relationship among these 5 families. The marriage was between the Barbadori and Castellan families. This adjacency matrix represents an *undirected network* because the matrix contains no directionality. We could add directionality by incorporating which family proposed a marriage if such information were available. In contrast, the Twitter data we analyze later are an example of a *directed network* where any relationship between a pair of units specifies a *sender* and a *receiver*. For an undirected network, the adjacency matrix is symmetric: the (i, j) element has the same value as the (j, i) element. Finally, using the `rowSums()` or `colSums()` function, we can check which family had the largest number of marriage relationships.

```
rowSums(florence)
## ACCIAIUOL    ALBIZZI BARBADORI   BISCHERI CASTELLAN     GINORI
##         1          3         2          3         3          1
##  GUADAGNI LAMBERTES     MEDICI      PAZZI   PERUZZI      PUCCI
##         4          1         6          1         3          0
##   RIDOLFI  SALVIATI    STROZZI TORNABUON
##         3          2         4          3
```

The result shows that the Medici family had 6 marriage relationships. It turns out that through this marriage network, the Medici family made themselves the most powerful faction in Renaissance Florence and eventually took over the state.

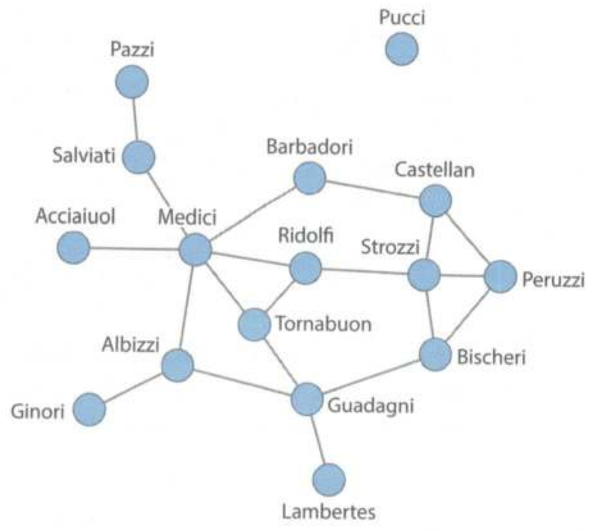
Network data carry information about the relationships between units. A **directed network** contains directionality, with senders and receivers, whereas an **undirected network** does not. An **adjacency matrix**, whose entries indicate the existence or absence of a relationship between two units, is one way to represent network data. An undirected network yields a symmetric adjacency matrix, whereas a directed network does not.

5.2.2 UNDIRECTED GRAPH AND CENTRALITY MEASURES

The most common tool for visualizing network data is a *graph*, which is also a mathematical object, as well as a visualization tool. A graph $\mathcal{G}$ consists of a set of *nodes* (or *vertices*) V and a set of *edges* (or *ties*) E, i.e., $\mathcal{G} = (V, E)$. A node represents an individual unit, or a family in our current example, and is typically depicted as a solid circle. An edge, on the other hand, represents the existence of a relationship between any pair of nodes (e.g., a marriage relationship between two families) via a line connecting those nodes.

The **igraph** package makes it easy to visualize network data as a graph. Be sure to install the package if you have not done so already. We first use the `graph.adjacency()` function to turn an adjacency matrix into an `igraph` object, meaning an object that the **igraph** package can use. We set the `mode` argument to `"undirected"` since we are analyzing an undirected network. We also specify `diag = FALSE` to indicate the assumption that there is no marriage within a family, resulting in a value of zero for every diagonal element of the adjacency matrix. Finally, we can visualize the marriage network data as a graph by applying the `plot()` function to the igraph object.

```
library("igraph")  # load the package
florence <- graph.adjacency(florence, mode = "undirected", diag = FALSE)
plot(florence) # plot the graph
```



The Medici family appears to occupy the center of the Florentine marriage network, being connected to various parts of the graph. We now introduce a variety of graph-based measures that can quantify *centrality*, or the extent to which each node is connected to other nodes and appears in the center of a graph. The number of edges, or *degree*, is perhaps the most crude measure of how well a node is connected to the other nodes in a graph. Figure 5.2a illustrates this measure using a simple undirected network example, where degree is indicated as an integer value within each node. Above, we found that the Medici family had the largest number of marriage relationships, so it has the highest degree. The degree of every node can be calculated by applying the `degree()` function to the `igraph` object.

```
degree(florence)

## ACCIAIUOL    ALBIZZI BARBADORI  BISCHERI CASTELLAN    GINORI
##         1          3         2         3         3         1
##  GUADAGNI LAMBERTES     MEDICI     PAZZI   PERUZZI     PUCCI
##         4          1         6         1         3         0
##   RIDOLFI  SALVIATI    STROZZI TORNABUON
##         3          2         4         3
```

Degree is problematically a local measure because it simply counts the number of edges that come out of a given node. As a result, it does not account for the structure of the graph beyond its immediate neighbors. As an alternative, we can count the sum of edges from a given node to all other nodes in a graph, including the ones that are not directly connected. This measure, called *farness*, describes how far apart a given node is from each one of all other nodes in the graph. This contrasts with degree, which counts the number of connected nodes. The inverse of farness, *closeness*, represents another

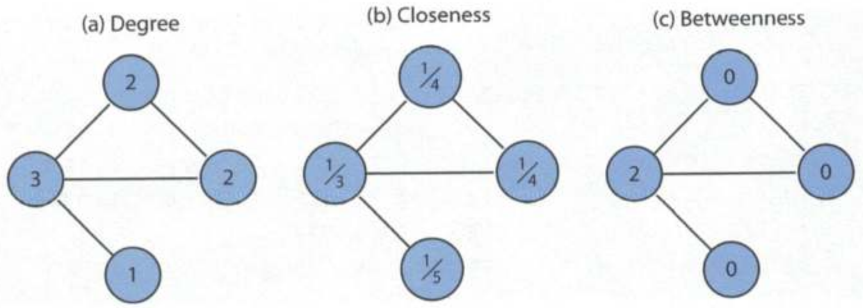


Figure 5.2. Degree, Closeness, and Betweenness in an Undirected Network. This simple example of an undirected network illustrates three alternative measures of centrality: degree, closeness, and betweenness.

measure of centrality. The closeness for node v is defined as

$$\begin{aligned}\text{closeness}(v) &= \frac{1}{\text{farness}(v)} \\ &= \frac{1}{\sum_{u \in V,\ u \neq v} \text{distance between } v \text{ and } u},\end{aligned}$$

where the summation is taken over all nodes other than v itself. The distance between two nodes is the number of edges in the shortest path, which is the shortest sequence of connected nodes, between the two nodes of interest. Figure 5.2b shows the values of this measure for each node in a simple example of an undirected network. In R, we can use the `closeness()` function to compute this measure. If a node is not connected to any other node, then the number of nodes in the graph, i.e., 16 in this case, is used instead of the length of the shortest path. Thus, Pucci family's closeness is equal to $1/(15 \times 16)$.

```
closeness(florence)

##   ACCIAIUOL     ALBIZZI   BARBADORI    BISCHERI   CASTELLAN
## 0.018518519 0.022222222 0.020833333 0.019607843 0.019230769
##      GINORI    GUADAGNI   LAMBERTES      MEDICI       PAZZI
## 0.017241379 0.021739130 0.016949153 0.024390244 0.015384615
##     PERUZZI       PUCCI    RIDOLFI    SALVIATI     STROZZI
## 0.018518519 0.004166667 0.022727273 0.019230769 0.020833333
##   TORNABUON
## 0.022222222
```

As with degree, we find that the Medici family has the largest value of closeness. To facilitate the interpretation of this measure, we can calculate the average number of edges from a given node to another node. This is done by dividing the farness by the number of other nodes on a graph. In the current example, we have a total of 16 nodes and so we divide the farness by 15. The results below imply that on average, there are 2.7 edges between the Medici family and another family in this network, which is the lowest among all families considered in this network data.

```
1 / (closeness(florence) * 15)
## ACCIAIUOL   ALBIZZI BARBADORI  BISCHERI CASTELLAN    GINORI
##  3.600000  3.000000  3.200000  3.400000  3.466667  3.866667
##  GUADAGNI LAMBERTES    MEDICI     PAZZI   PERUZZI     PUCCI
##  3.066667  3.933333  2.733333  4.333333  3.600000 16.000000
##   RIDOLFI  SALVIATI   STROZZI TORNABUON
##  2.933333  3.466667  3.200000  3.000000
```

A different type of centrality measure is *betweenness*. According to this measure, a node is considered to be central if it is responsible for connecting other nodes. In particular, if we assume that communication between a pair of nodes occurs through the shortest path between them, a node that lies on many such shortest paths may possess special leverage within a network. For a given node v, we calculate betweenness in three steps. First, compute the proportion of the shortest paths between two other nodes, t and u, that contain v. For example, two shortest paths occur between the Albizzi family and Tornabuon family, but we want only the proportion that contain v. Second, calculate this proportion for every unique pair of nodes t and u in the graph, excluding v. Third, sum all proportions. The formal definition is given by

$$\text{betweenness}(v) = \sum_{(t,u)\in V,\ t\neq v,\ u\neq v} \frac{\text{number of shortest paths that contain node } v}{\text{number of shortest paths between nodes } t \text{ and } u}.$$

Figure 5.2c illustrates this centrality measure in the same undirected network example used for the other two measures.

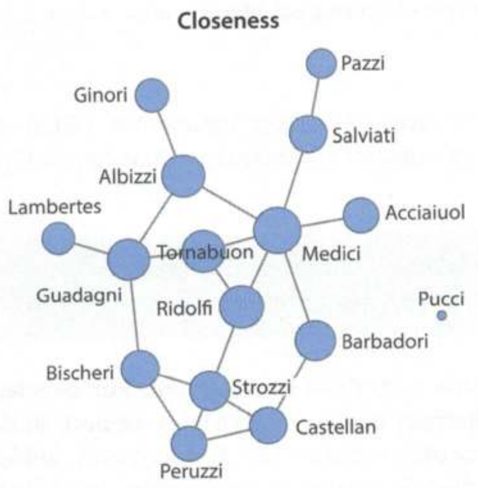
The `betweenness()` function can be used to compute this measure. We find that by far, the Medici family has the highest value of betweenness. In fact, since any given node can be uniquely paired with 105 other nodes, the Medici family lies in the shortest path of more than 45% of all possible pairs of other nodes.

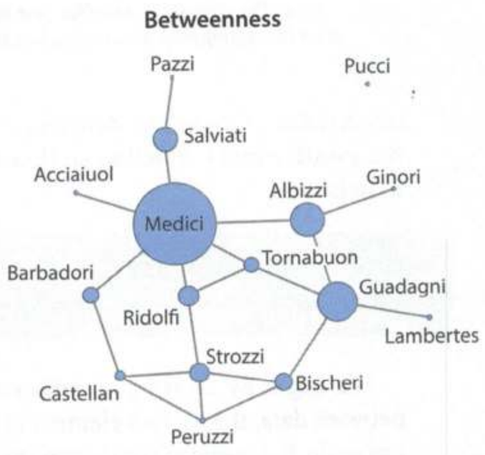
```
betweenness(florence)
## ACCIAIUOL   ALBIZZI BARBADORI  BISCHERI CASTELLAN    GINORI
##  0.000000 19.333333  8.500000  9.500000  5.000000  0.000000
##  GUADAGNI LAMBERTES    MEDICI     PAZZI   PERUZZI     PUCCI
## 23.166667  0.000000 47.500000  0.000000  2.000000  0.000000
##   RIDOLFI  SALVIATI   STROZZI TORNABUON
## 10.333333 13.000000  9.333333  8.333333
```

A **graph** is another way to represent network data where nodes (vertices) represent units and an edge (or tie) between two nodes indicates that a relationship exists between them. There are various **centrality** measures, including degrees, closeness, and betweenness. These measures evaluate the extent to which each node plays a central role in a graph.

We visualize the Florentine marriage network data by making the size of each node proportional to two centrality measures, closeness and betweenness. The values of closeness are relatively small, and so we multiply them by 1000 in order to enlarge the nodes of the graph.

```
plot(florence, vertex.size = closeness(florence) * 1000,
     main = "Closeness")
plot(florence, vertex.size = betweenness(florence),
     main = "Betweenness")
```





The graphs illustrate that the Medici family stands out especially in terms of betweenness, while the closeness measure suggests they are one of several well-connected families. In sum, using three measures of centrality—degree, closeness, and betweenness—we find that the Medici family is the most connected and central in the network of Florentine marriage relationships. In Renaissance Florence, the Medici family had the largest number of marriage relationships, was closely connected to other families, and occupied a critical position in marriages among other families.

5.2.3 TWITTER-FOLLOWING NETWORK

The Florentine marriage network data exemplify an *undirected* network where each edge has no directionality. Next, we analyze Twitter-following data among US senators as *directed* network data. In this data set, an edge represents an instance of a senator following another senator's Twitter account.[8] The data consist of two files, one listing pairs of the Twitter screen names of the following and followed politicians, `twitter-following.csv`, and the other containing information about each politician, `twitter-senator.csv`. Table 5.4 lists the names and descriptions

[8] This data set is generously provided by Pablo Barberá.

Table 5.4. Twitter Following Data.

Variable	*Description*
Twitter-following data	
`following`	Twitter screen name of the following senator
`followed`	Twitter screen name of the followed senator
Twitter senator data	
`screen_name`	Twitter screen name
`name`	name of senator
`party`	party (D = Democrat, R = Republican, I = Independent)
`state`	state abbreviation

Note: The data are in two files, one listing the pairs of following and followed senators and the other containing information about each senator.

of variables in these two data files. We set the stringAsFactors argument to FALSE in the `read.csv()` function so that names of senators are treated as characters rather than factors.

```r
twitter <- read.csv("twitter-following.csv", stringsAsFactors = FALSE)
senator <- read.csv("twitter-senator.csv", stringsAsFactors = FALSE)
```

We begin by creating an adjacency matrix with these two data sets. For directed network data, the (i, j)th element of the adjacency matrix is 1 if an edge connects node i to node j. A value of 0 indicates the absence of any relationship. Consequently, unlike the case of undirected network data, the adjacency matrix is asymmetric: the (i, j)th element of this matrix may not equal its (j, i)th element. We create this adjacency matrix by initializing it with a matrix of zeros and then changing the value of its (i, j)th element from 0 to 1 if the ith politician follows the jth politician.

```r
n <- nrow(senator) # number of senators
## initialize adjacency matrix
twitter.adj <- matrix(0, nrow = n, ncol = n)
## assign screen names to rows and columns
colnames(twitter.adj) <- rownames(twitter.adj) <- senator$screen_name
## change "0" to "1" when edge goes from node "i" to node "j"
for (i in 1:nrow(twitter)) {
    twitter.adj[twitter$following[i], twitter$followed[i]] <- 1
}
```

Finally, as before, we use the `graph.adjacency()` function to turn the adjacency matrix into an `igraph` object. This time, however, we need to specify its mode argument as `"directed"` to indicate that the input is a directed network data set.

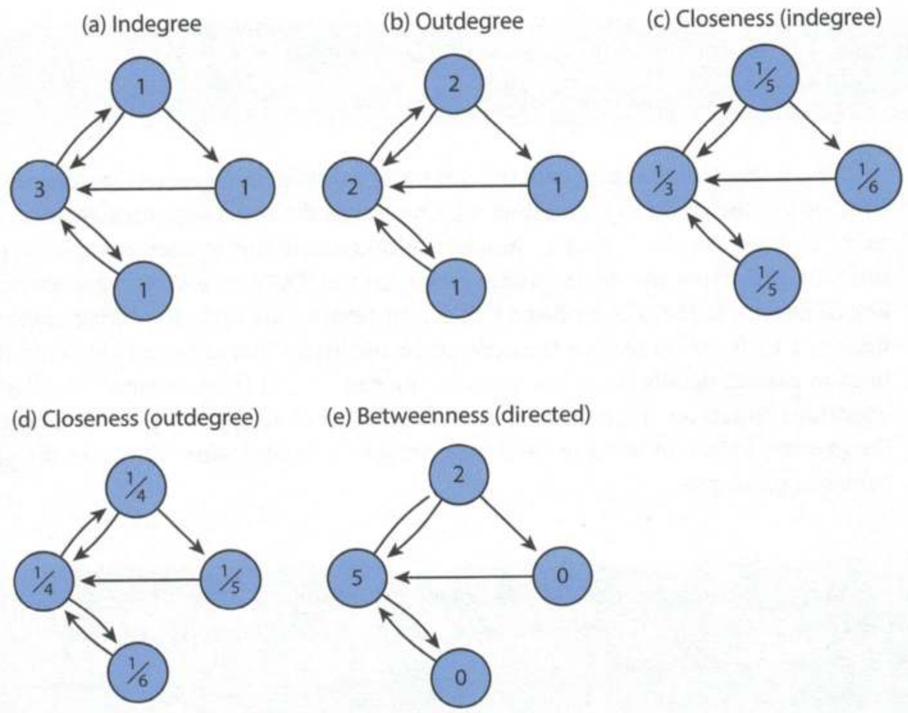


Figure 5.3. Degree, Closeness, and Betweenness in a Directed Network. This simple example of directed network data illustrates three alternative measures of centrality: degree, closeness, and betweenness.

```
twitter.adj <- graph.adjacency(twitter.adj, mode = "directed", diag = FALSE)
```

5.2.4 DIRECTED GRAPH AND CENTRALITY

We can define the three centrality measures discussed earlier for a directed network. We now have two types of *degree* measures. The sum of edges coming to a node (i.e., the number of times a politician's Twitter account is followed by another politician) is called *indegree*, whereas the sum of edges coming out of a node (i.e., the number of times a politician follows the Twitter account of another politician) is called *outdegree*. Figures 5.3a and 5.3b illustrate the two degree measures using a simple directed network. The `degree()` function accepts an argument `mode` with three options, `"in"` for indegree, `"out"` for outdegree, and `"total"` (the default when `mode` is unspecified) for total degree, which is the sum of indegree and outdegree. We compute and store indegree and outdegree as additional variables in the `senator` data frame. By construction, the `twitter.adj` matrix has the same ordering of senators as the `senator` data frame. As a result, one can insert the output of the `degree()` function without sorting them.

```
senator$indegree <- degree(twitter.adj, mode = "in")
senator$outdegree <- degree(twitter.adj, mode = "out")
```

Next, we extract the cases with the 3 greatest values of indegree and outdegree. To do this, we use the `order()` function, which returns the ordering index vector. Like the `sort()` function, the `order()` function allows one to sort in decreasing or increasing order by specifying the `decreasing` argument as `TRUE` or `FALSE`, respectively. The key difference is that the `order()` function returns the ordering index vector while the `sort()` function returns the ordered vector itself. This ordering index can then be used to extract details about the cases of interest. Recall from section 3.7.2 that the `$` operator extracts an element from a list. Below, we identify the 3 politicians who have the greatest values of indegree and another set of 3 politicians who have the greatest values of outdegree.

```
in.order <- order(senator$indegree, decreasing = TRUE)
out.order <- order(senator$outdegree, decreasing = TRUE)
## 3 greatest indegree
senator[in.order[1:3], ]

##        screen_name             name party state indegree
## 51   SenJohnMcCain      John McCain     R    AZ       64
## 57   lisamurkowski   Lisa Murkowski     R    AZ       60
## 18 SenatorCollins Susan M. Collins     D    ME       58
##    outdegree
## 51        15
## 57        87
## 18        79

## 3 greatest outdegree
senator[out.order[1:3], ]

##        screen_name                  name party state indegree
## 37   SenDeanHeller           Dean Heller     R    NV       55
## 21     SenBobCasey Robert P. Casey, Jr.     D    PA       43
## 65 sendavidperdue          David Perdue     R    GA       30
##    outdegree
## 37        89
## 21        88
## 65        88
```

The other two measures of centrality introduced above, *closeness* and *betweenness*, can be defined for directed network data as well. There are three ways to define a path from one node to another. We can ignore directionality as in the case of undirected networks or incorporate it in one of two ways: traveling along an outgoing path in its direction, or traveling along an incoming path against its direction. Closeness for incoming paths corresponds to indegree, while closeness for outgoing paths corresponds to outdegree. Figures 5.3c and 5.3d illustrate the closeness measures based on incoming and outgoing paths, respectively.

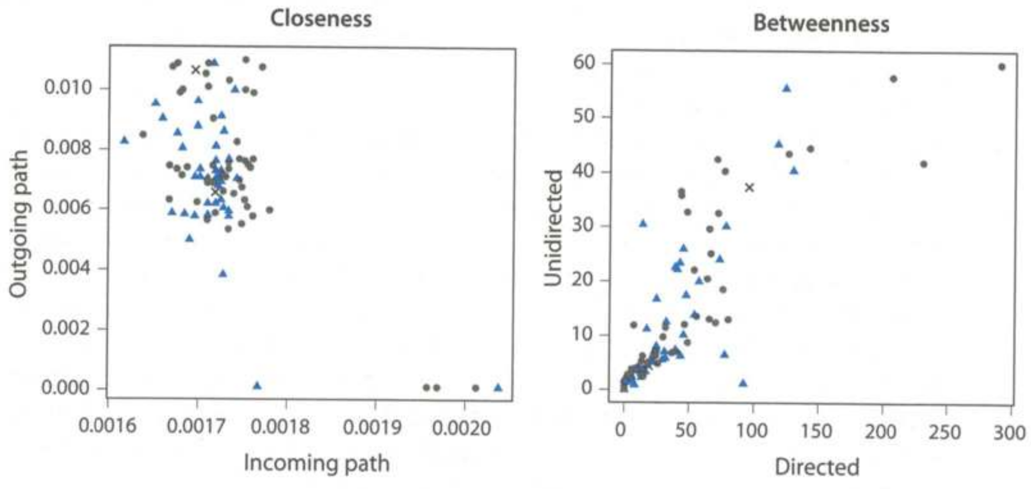
To compute closeness in R, therefore, the `closeness()` function has the `mode` argument, which can take either `"in"` (incoming path), `"out"` (outgoing path), or `"total"` (ignore directionality). Betweenness, however, sees only two options (`direct = TRUE` or `FALSE`), because the distinction between incoming and outgoing paths does not make sense from the perspective of a node in the path between two other nodes (see figure 5.3e). In particular, the `betweenness()` function takes a logical value for the `directed` argument, indicating whether to consider directionality. Below, we first graphically compare two closeness measures (incoming versus outgoing path) and then compare directed betweenness against undirected betweenness using another plot. Before making these plots, we set the parameters for colors and symbols based on party. Specifically, we use blue triangles for Democrats, red circles for Republicans, and black crosses for Independents.

```
n <- nrow(senator)
## color: Democrats = blue, Republicans = red, Independent = black
col <- rep("red", n)
col[senator$party == "D"] <- "blue"
col[senator$party == "I"] <- "black"
## pch: Democrats = triangle, Republicans = circle, Independent = cross
pch <- rep(16, n)
pch[senator$party == "D"] <- 17
pch[senator$party == "I"] <- 4
```

Using these color and symbol parameters, we are now ready to make the plots.

```
## plot for comparing two closeness measures (incoming vs. outgoing)
plot(closeness(twitter.adj, mode = "in"),
     closeness(twitter.adj, mode = "out"), pch = pch,  col = col,
     main = "Closeness", xlab = "Incoming path", ylab = "Outgoing path")
## plot for comparing directed and undirected betweenness
plot(betweenness(twitter.adj, directed = TRUE),
     betweenness(twitter.adj, directed = FALSE), pch = pch, col = col,
     main = "Betweenness", xlab = "Directed", ylab = "Undirected")
```

The plots below use solid gray circles for Republicans instead of red gray circles. See page C4 for the full-color version.



There is little association between the two closeness measures based on incoming and outgoing paths. This suggests that in this Twitter network, a senator's closeness to other senators in terms of being followed by them has little relationship to closeness based on the same senator following them. Interestingly, however, the betweenness measure is quite similar, regardless of whether one incorporates directionality. The two betweenness measures suggest that several Republican senators are well connected and central to the network (see the upper-right corner of the right-hand plot).

As a final alternative measure of centrality, we introduce *PageRank*. PageRank was developed by the cofounders of *Google*, Sergey Brin and Larry Page, to optimize the ranking of websites for their search engine outcomes. PageRank is computed using an iterative algorithm. In section 3.7, we saw k-means clustering as an example of an iterative algorithm. PageRank is based on the idea that nodes with a greater number of incoming edges are more important. Intuitively, we can think of incoming edges as votes of support. In the Twitter example, those senators who have a large number of followers are seen as more important. Furthermore, if a node has an incoming edge from another node with a large number of incoming edges, it results in a greater value of PageRank than if it has an incoming edge from a node with fewer incoming edges. In other words, if the Twitter account of a politician is followed by another politician whose account has many followers, they receive a larger PageRank than they would if followed by a politician with fewer followers. Finally, we note that the sum of PageRank values across all nodes equals 1.

The algorithm begins by assigning a set of initial PageRank values to all nodes. At each iteration, the PageRank value for node j will be updated using

$$\text{PageRank}_j = \frac{1-d}{n} + d \times \sum_{i=1}^{n} \underbrace{\frac{A_{ij} \times \text{PageRank}_i}{\text{outdegree}_i}}_{\text{"vote" from node } i \text{ to node } j} . \qquad (5.2)$$

In this equation, A_{ij} is the (i, j)th element of the adjacency matrix indicating whether or not an edge connects node i to node j, d is a constant to be specified (typically set to 0.85), and n is the number of nodes. The equation shows that the PageRank for a given node j equals the sum of "votes" from other nodes that have an incoming edge into node j. If there is no edge from node i to node j, then $A_{ij} = 0$, and therefore no vote is given to node j from node i. However, if $A_{ij} = 1$, then a vote from node i to node j is equal to the PageRank value of node i divided by node i's outdegree. This means that each node must equally allocate its PageRank value across all other nodes to which it has outgoing edges. For example, if a node has a PageRank value of 0.1 and has two outgoing edges, then each receiver obtains 0.05 from this node. This iterative algorithm stops when the PageRank values for all nodes no longer change.

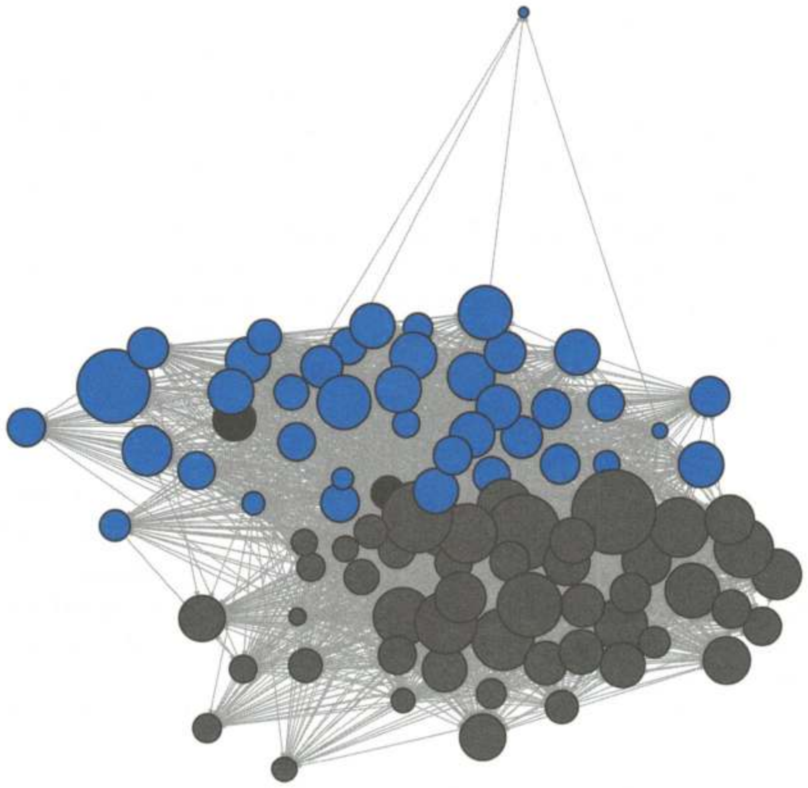
There are several **centrality measures** for **directed networks**, including indegree and outdegree, closeness (based on incoming edges, outgoing edges, or both), and betweenness (with or without directionality). **PageRank** is an iterative algorithm that produces a centrality measure where each node equally allocates its "votes" to other connected nodes.

In R, we can compute PageRank by the `page.rank()` function. The function can also be applied to an undirected network by setting the `directed` argument to `FALSE` (the default value is `TRUE`). The output object is a list that includes a numeric vector of PageRank, as an element called `vector`.

```
senator$pagerank <- page.rank(twitter.adj)$vector
```

Below, we visualize usage of the Twitter network among US senators by setting node size proportional to PageRank. The `plot()` function for adjacency matrices takes several arguments, including `vertex.size` (to adjust the size of each node), `vertex.color` (to adjust the color of each node), `vertex.label` (to specify the label of each node), `edge.arrow.size` (to adjust the size of each edge's arrow), and `edge.width` (to adjust the width of each edge). See `?igraph.plotting` for more details.

```
## "col" parameter is defined earlier
plot(twitter.adj, vertex.size = senator$pagerank * 1000,
     vertex.color = col, vertex.label = NA,
     edge.arrow.size = 0.1, edge.width = 0.5)
```

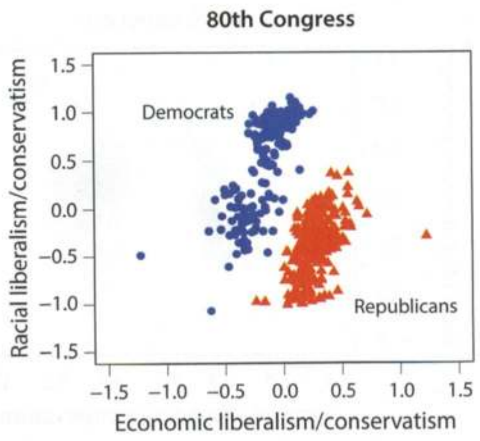


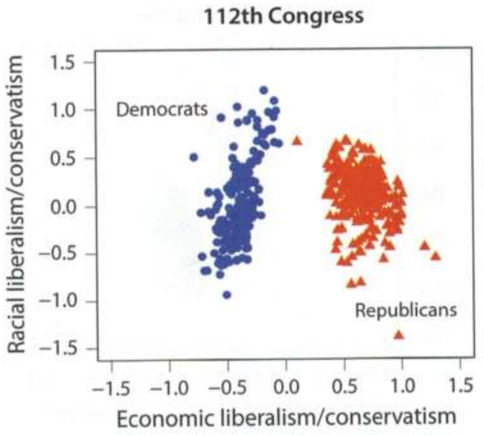
The plot above shows that this Twitter network is quite dense but also polarized. Republican senators appear to be clustered together while Democrats form another cluster, suggesting that senators tend to follow senators of their own party. In addition, while Republican senators appear to have slightly greater PageRank values than Democrats, the partisan difference is minor.

To better understand the algorithm, we consider a function that updates the PageRank at each iteration according to equation (5.2). Let n be the number of nodes in a graph, A be an n × n adjacency matrix, d be a constant, and pr be a vector of PageRank values from the previous iteration. Then, this function can be defined as follows.

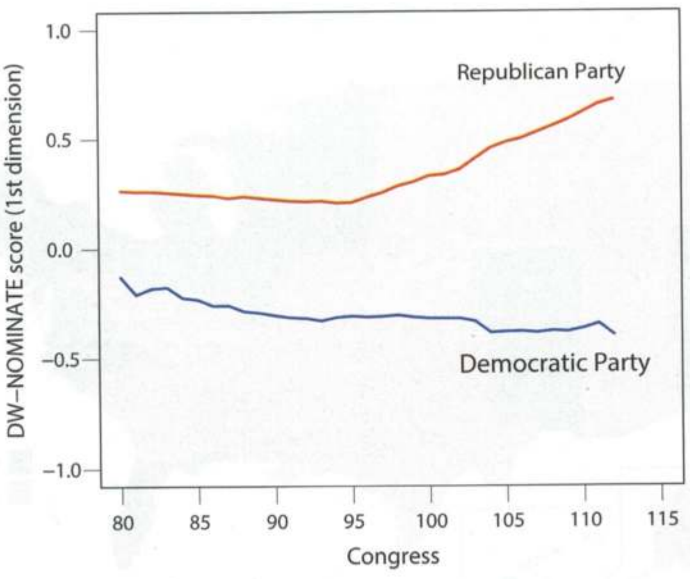
```
PageRank <- function(n, A, d, pr) { # function takes 4 inputs
    deg <- degree(A, mode = "out") # outdegree calculation
    for (j in 1:n) {
        pr[j] <- (1 - d) / n +  d * sum(A[ ,j] * pr / deg)
    }
    return(pr)
}
```

We will apply this function to the simple network used in figure 5.3. We will use the while() loop so that the algorithm stops when the differences in PageRank

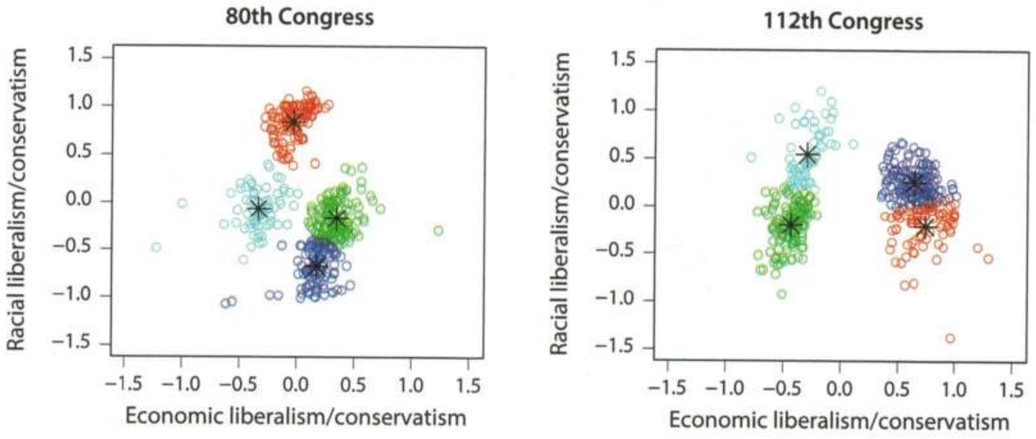




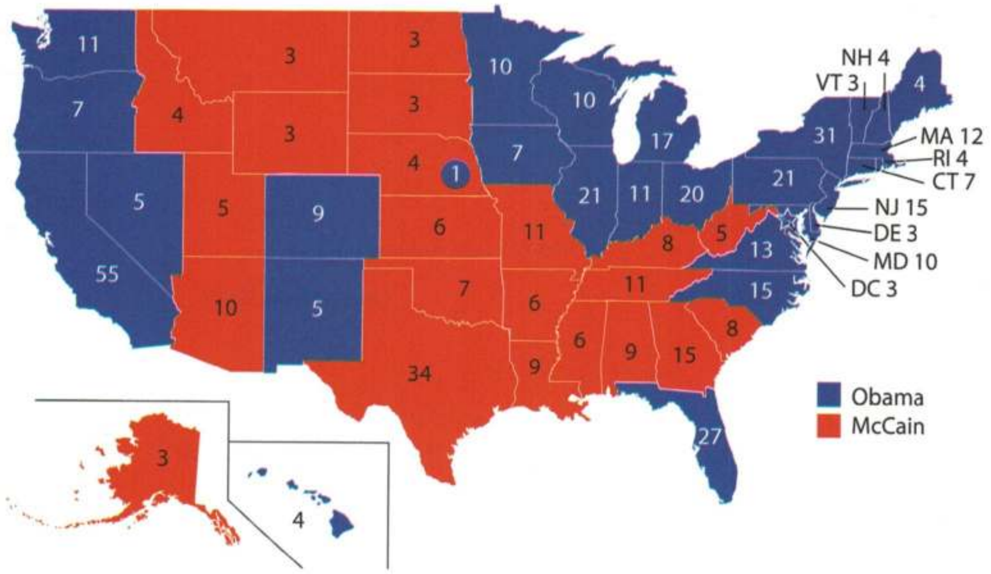
The full-color version of the plots on page 99 in section 3.6.1.



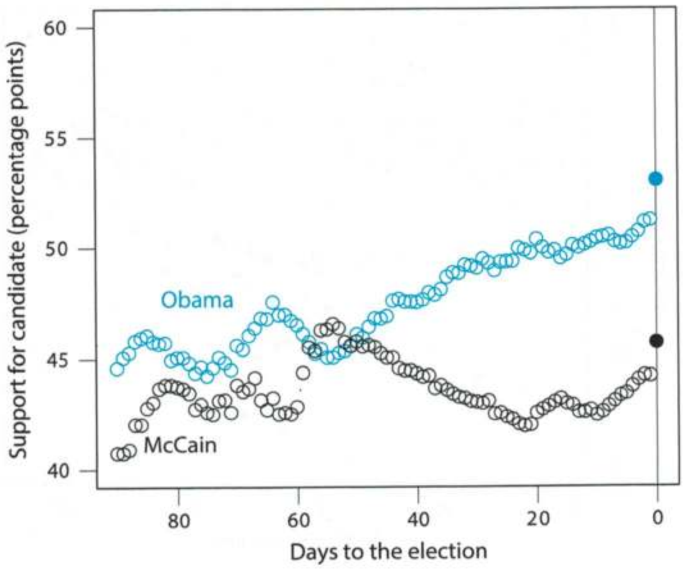
The full-color version of the plot on page 100 in section 3.6.1.



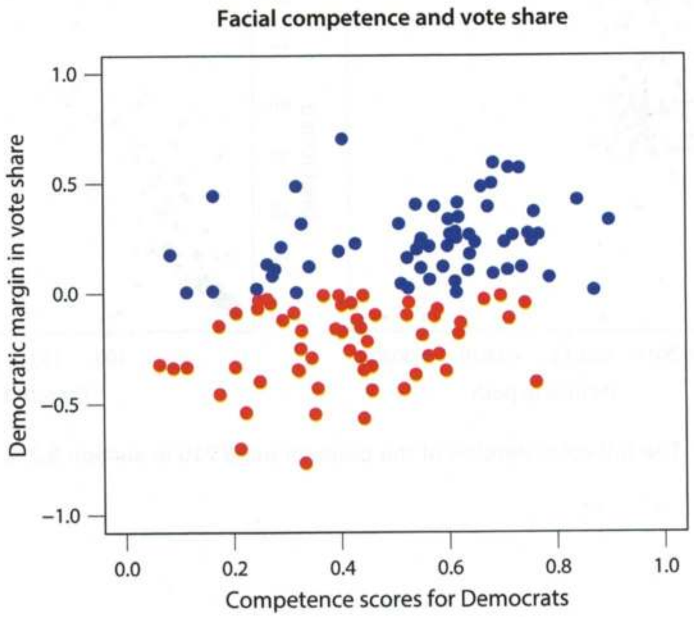
The full-color version of the plots on page 114 in section 3.7.3.



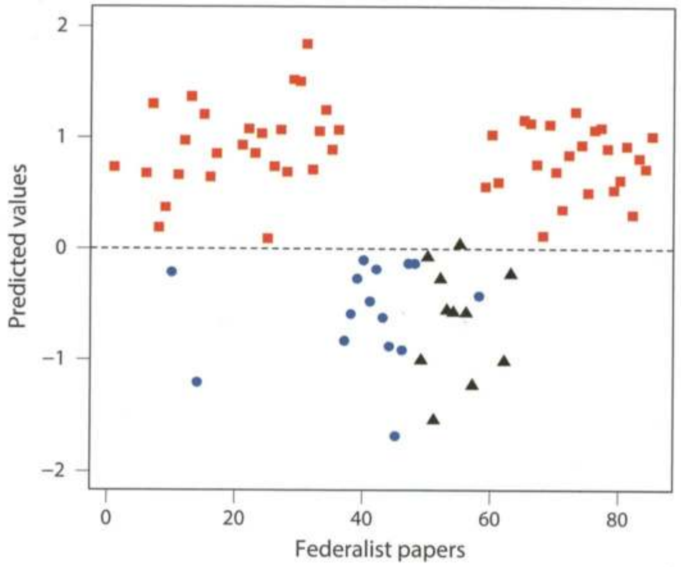
The full-color version of figure 4.1 on page 124.



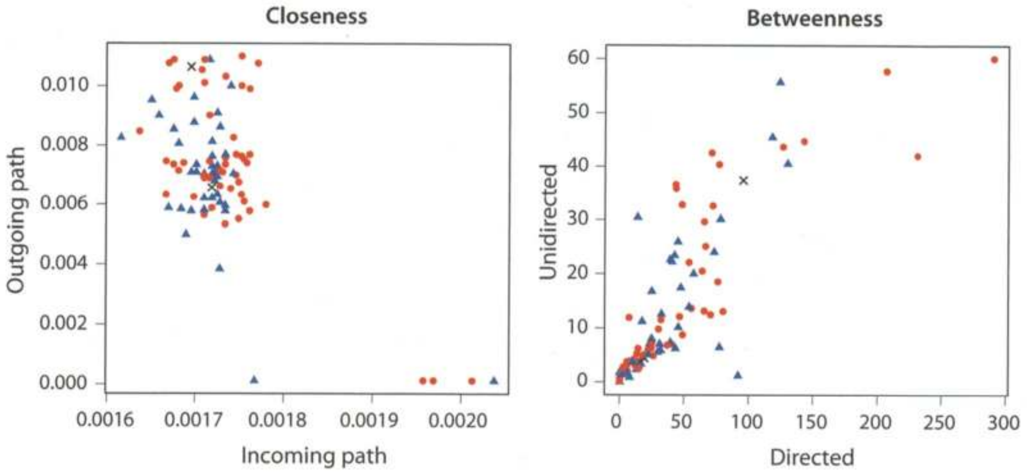
The full-color version of the plot on page 138 in section 4.1.3.



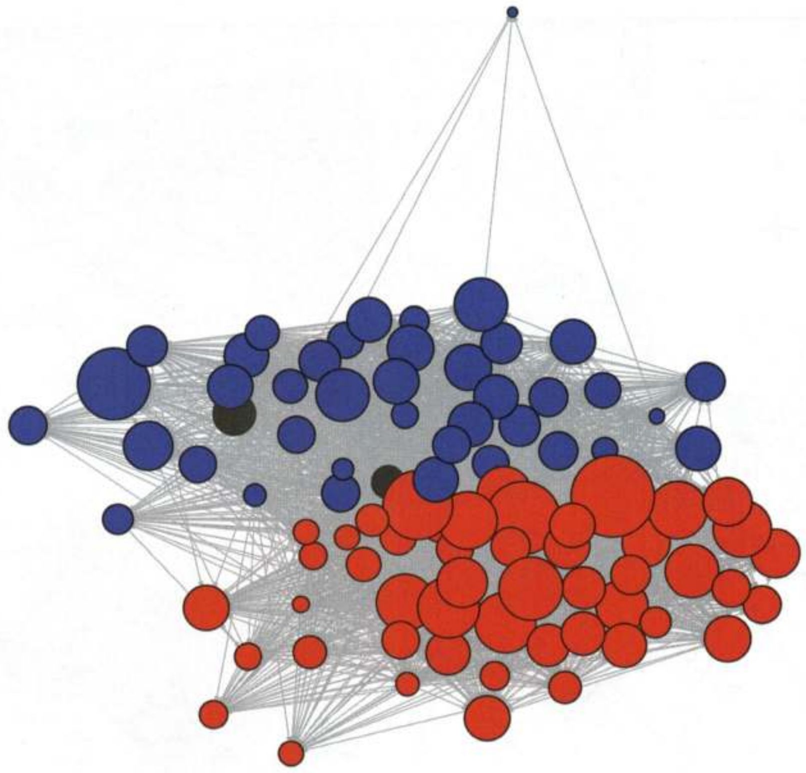
The full-color version of the plot on page 141 in section 4.2.1.

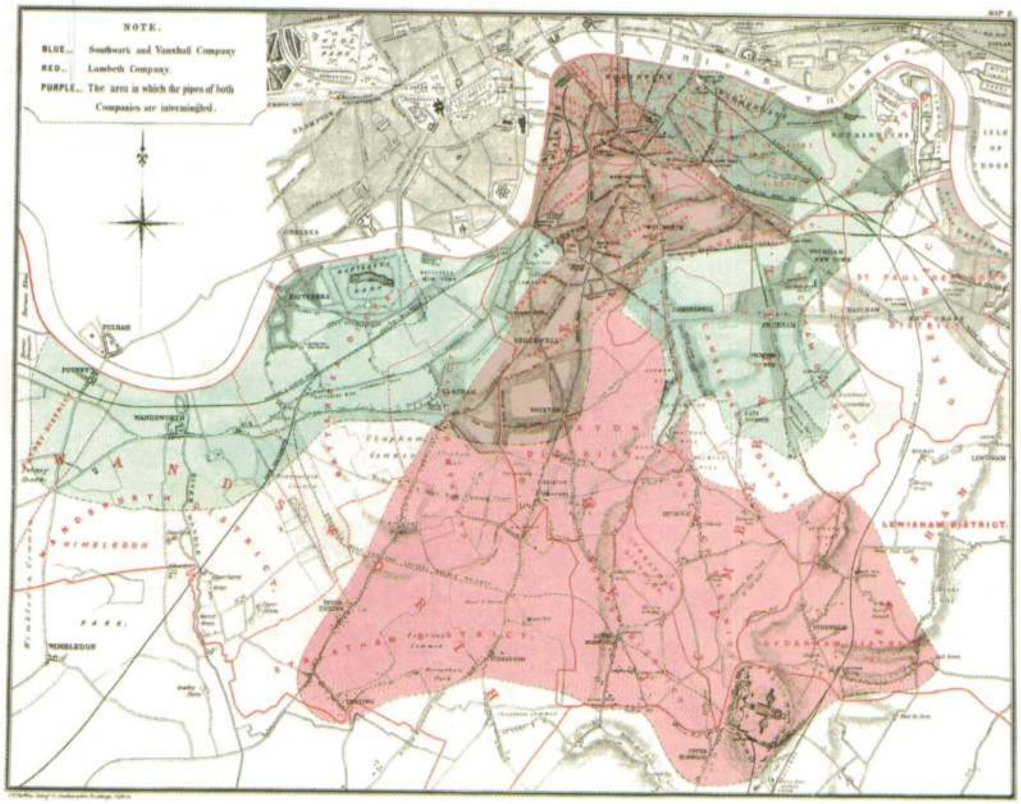


The full-color version of the plot on page 205 in section 5.1.5.

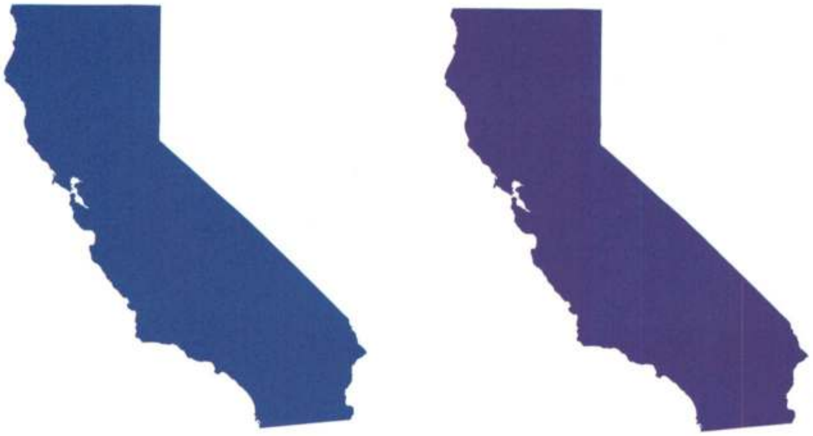


The full-color version of the plots on page 216 in section 5.2.4.

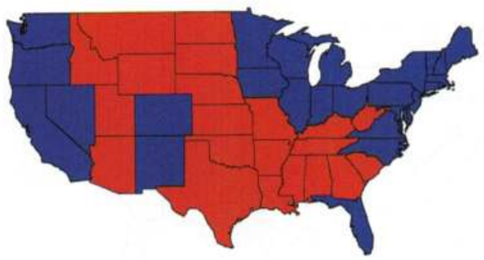
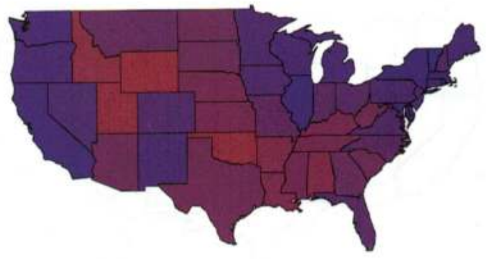

The full-color version of the plot on page 218 in section 5.2.2.



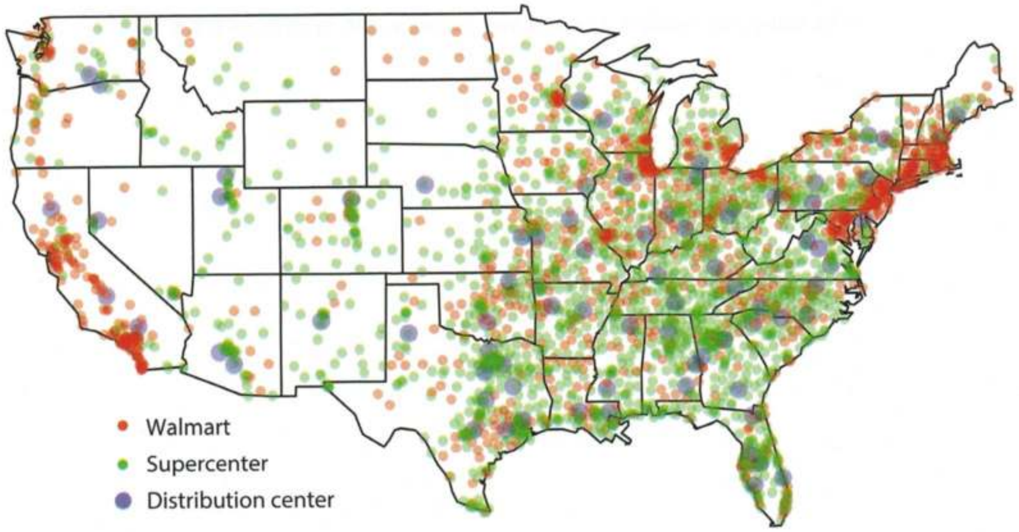
The full-color version of figure 5.5 on page 222.



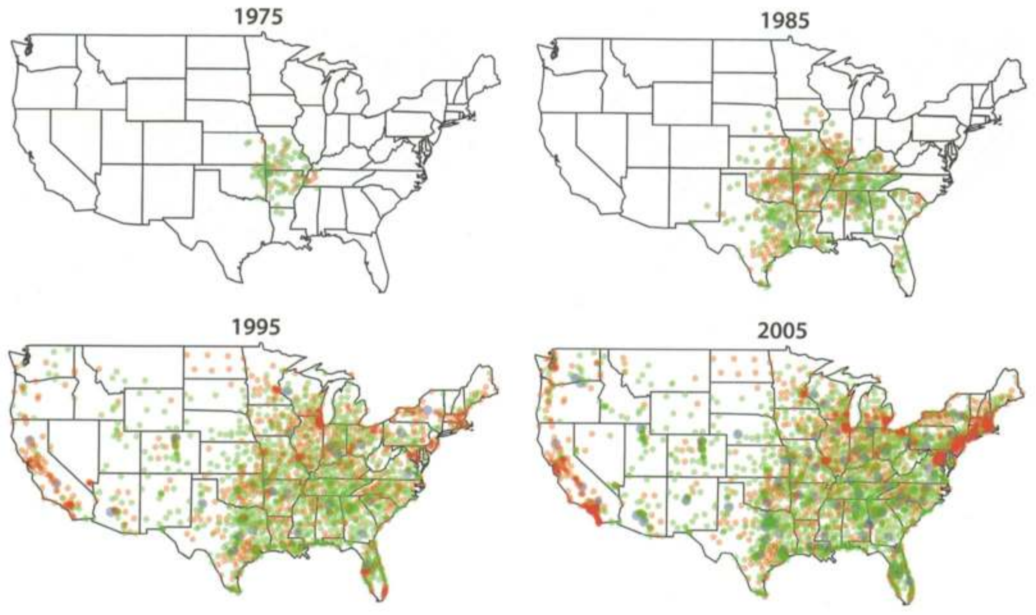
The full-color version of the maps on page 229 in section 5.3.4.

The full-color version of the maps on page 230 in section 5.3.4.



The full-color version of the map on page 232 in section 5.3.4



The full-color version of the maps on page 234 in section 5.3.6.

values between two successive iterations become negligible. The `while` loop takes the syntax

```
while (condition) {

    LOOP CONTENTS HERE

}
```

where the loop contents will be executed repeatedly so long as the conditional statement, `condition`, is evaluated to be `TRUE`. In our application, we will compute the maximum absolute difference in PageRank values between two successive iterations and stop the algorithm when this becomes less than a prespecified threshold. To test this script, we first construct an adjacency matrix with arbitrary values.

```
nodes <- 4
## adjacency matrix with arbitrary values
adj <- matrix(c(0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0),
              ncol = nodes, nrow = nodes, byrow = TRUE)
adj
##      [,1] [,2] [,3] [,4]
## [1,]    0    1    0    1
## [2,]    1    0    1    0
## [3,]    0    1    0    0
## [4,]    0    1    0    0
adj <- graph.adjacency(adj)  # turn it into an igraph object
```

To implement the PageRank algorithm, we set the starting values and specify the constant *d* in the algorithm (we choose 0.85). We then use the `while()` loop to iteratively run the algorithm until a convergence criterion is satisfied. For the convergence criterion, we use 0.001 as the threshold for the maximal absolute difference in the PageRank values between two successive iterations. We use equal PageRank values across nodes as their starting values.

```
d <- 0.85  # typical choice of constant
pr <- rep(1 / nodes, nodes) # starting values
## maximum absolute difference; use a value greater than threshold
diff <- 100
## while loop with 0.001 being the threshold
while (diff > 0.001) {
    pr.pre <- pr # save the previous iteration
    pr <- PageRank(n = nodes, A = adj, d = d, pr = pr)
    diff <- max(abs(pr - pr.pre))
}
```

```
pr

## [1] 0.2213090 0.4316623 0.2209565 0.1315563
```

The result shows that the second observation has the highest PageRank value. This makes sense because, as shown in the adjacency matrix, this observation has the greatest number of incoming edges, represented by the second column.

5.3 Spatial Data

In addition to texts and networks, we introduce another type of data, *spatial data*. Spatial data are best analyzed by visualization through *maps*. This chapter covers two types of spatial data. One is *spatial point data*, which can be plotted as a set of points on a map. The other is *spatial polygon data*, which represent a sequence of connected points on a map corresponding to the boundaries of certain areas such as counties, districts, and provinces. We also consider *spatial–temporal data*, which are a set of spatial point or polygon data recorded over time, revealing changes in spatial patterns over time.

5.3.1 THE 1854 CHOLERA OUTBREAK IN LONDON

In his book, *Mode of Communication of Cholera*, a British physician John Snow demonstrated the effective use of maps for visualizing the spatial distribution of fatal cholera cases. Snow collected the spatial point data about fatal cases in the Soho neighborhood of London during the 1854 outbreak and plotted this information on a map. Figure 5.4 reproduces the original map. Black rectangle areas indicate fatal cholera cases, which were found to cluster around the Broad Street water pump located at the center of the map. All water pumps are also indicated by solid circles and labeled as such on the map.

From this map, Snow discovered that fatal cholera cases were clustered on and around Broad Street. He speculated that cholera was spread by sewage-contaminated water, a theory the authorities and the water company were reluctant to believe. After extensive research that included close inspection of water and interviews with local residents, Snow concluded that the water pump at the corner of Broad and Cambridge Streets was the source of the cholera outbreak. He concluded by writing,

> The result of the inquiry then was, that there had been no particular outbreak or increase of cholera, in this part of London, except among the persons who were in the habit of drinking the water of the above-mentioned pump-well. (p. 40)

Snow also employed a "grand natural experiment" to show that the water supply of the Southwark and Vauxhall Company was responsible for the spread of cholera in London. Figure 5.5 reproduces the spatial polygon map that Snow used to visualize the area of the *natural experiment*, meaning a situation in the world that resembles an experiment without intervention from researchers. The map shows that the Lambeth

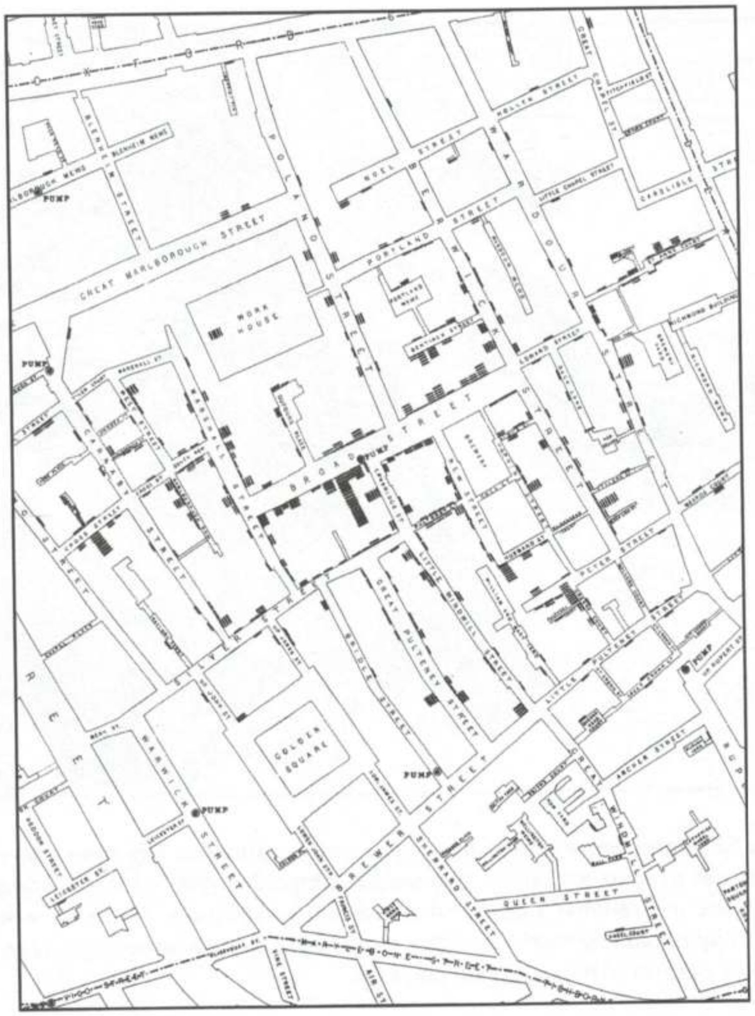


Figure 5.4. John Snow's Map of Fatal Cholera Cases in London. Black rectangle areas indicate fatal cholera cases, which were found to cluster around the Broad Street water pump. All water pumps are also indicated on the map. Original source: John Snow (1855) *Mode of Communication of Cholera*. London: John Churchill, New Burlington Street.

Company supplied cleaner water to the area further south (indicated by the red region), whereas the Southwark and Vauxhall Company provided contaminated water to the neighborhoods along the River Thames (indicated by the blue region). See page C6 for the full-color version. Snow argued that the overlapping area represented a natural experiment where two companies competed for customers: some people received their water supply from one company while their neighbors received water from the other company. Assuming that the two groups of customers were alike in all other respects, any difference in their cholera rates resulted from the choice of company.

After much research, Snow concluded that probably no *confounder* affected this *natural experiment*. Based on the discussion in section 2.5.2, confounding factors

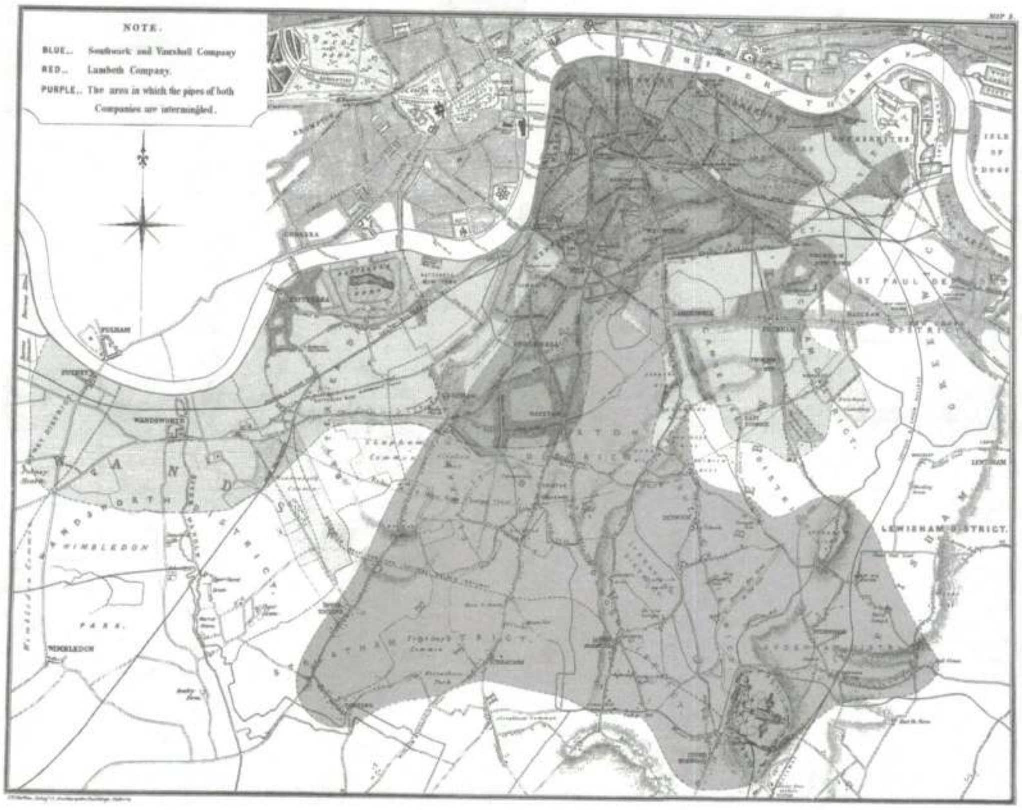

NOTE.
BLUE.. Southwark and Vauxhall Company
RED.. Lambeth Company.
PURPLE.. The area in which the pipes of both Companies are intermingled.
LEWISHAM DISTRICT.
WIMBLEDON

Figure 5.5. John Snow's Map of the Natural Experiment. The map shows the area of the natural experiment where two water companies (the Lambeth Company and the Southwark and Vauxhall Company) compete for customers. This area is represented by the overlap of red (Lambeth) and blue (Southwark and Vauxhall) regions as shown in the full-color version of this figure on page C6.

in this context refer to the variables associated with water companies and cholera outbreak rates of a neighborhood. He describes this experiment succinctly as follows:

> The mixing of the supply is of the most intimate kind. The pipes of each Company go down all the streets, and into nearly all the courts and alleys. A few houses are supplied by one Company and a few by the other, according to the decision of the owner or occupier at that time when the Water Companies were in active competition. In many cases a single house has a supply different from that on either side. Each Company supplies both rich and poor, both large houses and small; there is no difference either in the condition or occupation of the persons receiving the water of the different Companies....
>
> The experiment, too, was on the grandest scale. No fewer than three hundred thousand people of both sexes, of every age and occupation, and of every rank and station, from gentlefolks down to the very poor, were divided into two groups without their choice, and, in most cases, without their knowledge; one

group being supplied with water containing the sewage of London, and amongst it, whatever might have come from the cholera patients, the other group having water quite free from such impurity. (pp. 74–75)

By matching the addresses of persons dying of cholera with the companies that supplied them water, Snow was able to show that the overwhelming majority of deaths had occurred in the households with water supplied by the Southwark and Vauxhall Company.

Snow's book illustrates the power of spatial data analysis. In particular, the visualization of spatial data through maps enables researchers to discover previously unknown patterns and present their findings in a convincing manner.

5.3.2 SPATIAL DATA IN R

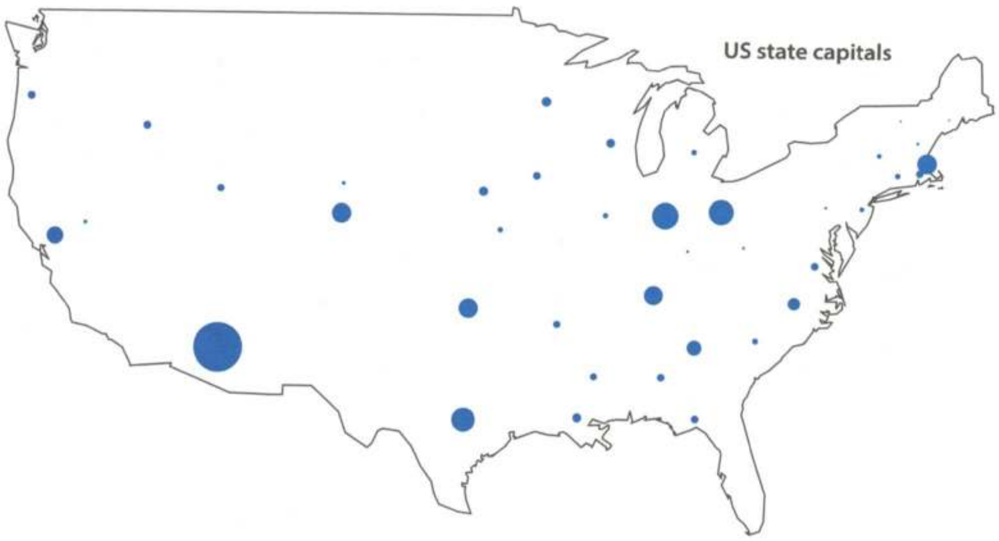
In chapter 4, we analyzed the 2008 US presidential election. Figure 4.1 presents a map of the Electoral College, efficiently visualizing the outcome of the election. This is an example of *spatial polygon data*, where each state represents a polygon whose boundaries can be constructed by connecting a series of points. We can then color each polygon or state blue (red) if Barack Obama (John McCain) won the plurality of votes within that state.

In R, the **maps** package provides various mapping tools as well as many spatial databases. The package contains a spatial database of various cities in the world. For example, it includes a data frame of US cities called `us.cities`. Any built-in data frame can be loaded by using the `data()` function. Below, we show the first few observations of this data set, which contains the name (as the `name` variable), state (`country.etc`), population (`pop`), latitude (`lat`), longitude (`long`), and whether the city is the capital of the country (`capital` = `1`), the capital of a state (`capital` = `2`), or neither (`capital` = `0`).

```
library(maps)
data(us.cities)
head(us.cities)
##           name country.etc    pop   lat    long capital
## 1 Abilene TX          TX 113888 32.45  -99.74       0
## 2   Akron OH          OH 206634 41.08  -81.52       0
## 3 Alameda CA          CA  70069 37.77 -122.26       0
## 4  Albany GA          GA  75510 31.58  -84.18       0
## 5  Albany NY          NY  93576 42.67  -73.80       2
## 6  Albany OR          OR  45535 44.62 -123.09       0
```

Now we can add state capitals to the map of the United States. We can use the `map()` function to access one spatial database and visualize the data therein. For example, in order to plot the United States, we set the `database` argument to `"usa"`. Spatial points can be easily added to maps using the `points()` function with their longitude and latitude information as the inputs for the x and y coordinates, respectively. Each state capital is represented by a solid circle whose size is proportional to its population. We can add a title by using the title() function after a map is drawn.

```
map(database = "usa")
capitals <- subset(us.cities, capital == 2) # subset state capitals
## add points proportional to population using latitude and longitude
points(x = capitals$long, y = capitals$lat, col = "blue",
       cex = capitals$pop / 500000, pch = 19)
title("US state capitals") # add a title
```



As another example, we plot the state of California. We use the "state" database, which contains the spatial polygon data about US states, and specify the regions argument to "California".

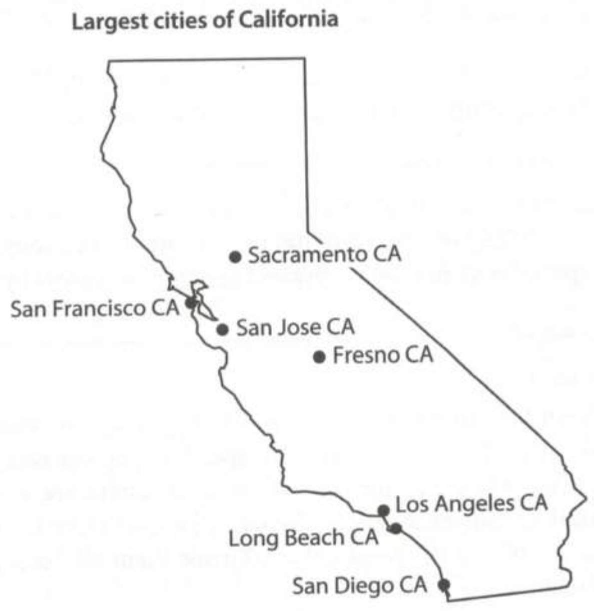
```
map(database = "state", regions = "California")
```

We will add to a map of California the seven cities that have the largest populations. To extract these cities from the data, we use the order() function as before (see section 5.2.4).

```
cal.cities <- subset(us.cities, subset = (country.etc == "CA"))
sind <- order(cal.cities$pop, decreasing = TRUE)# order by population
top7 <- sind[1:7] # seven cities with largest population
```

We now add these cities to the map using the `points()` function, as well as their city names using the `text()` function.

```
map(database = "state", regions = "California")
points(x = cal.cities$long[top7], y = cal.cities$lat[top7], pch = 19)
## add a constant to longitude to avoid overlapping with circles
text(x = cal.cities$long[top7] + 2.25, y = cal.cities$lat[top7],
     label = cal.cities$name[top7])
title("Largest cities of California")
```



It is instructive to consider what the spatial polygon data look like in R. To do this, we can set the `plot` argument of the `map()` function to `FALSE` to suppress the plotting. Then, the function will return a list object with a sequence of coordinates saved as `x` (x-coordinate or *longitude*) and `y` (y-coordinate or *latitude*). Within the list, `NA` separates different polygons whose names are stored as `names`. We use the US map to illustrate this.

```
usa <- map(database = "usa", plot = FALSE) # save map
names(usa)  # list elements

## [1] "x"      "y"      "range" "names"
```

Now, we can check the number of coordinates used to create the US map by computing the length of vector `x`. We also display the first few after combining the x- and y-coordinates into a matrix using the `cbind()` function.

```
length(usa$x)

## [1] 7252

head(cbind(usa$x, usa$y)) # first five coordinates of a polygon

##           [,1]     [,2]
## [1,] -101.4078 29.74224
## [2,] -101.3906 29.74224
## [3,] -101.3620 29.65056
## [4,] -101.3505 29.63911
## [5,] -101.3219 29.63338
## [6,] -101.3047 29.64484
```

We observe that the map of the United States consists of 7252 pairs of coordinates. The `map()` function connects these points to construct maps.

Spatial data contain information about patterns over space and can be visualized through maps. While **spatial point data** represent the locations of events as points on a map, **spatial polygon data** represent geographical areas by connecting points on a map.

5.3.3 COLORS IN R

We next learn how to color maps. Color is important for visualization in general, not simply for maps. So far, we have been specifying colors using names like `"red"` or `"blue"`. The only exception is section 3.7.3 where we used a set of integers that correspond to different colors through the `palette()` function. In fact, R knows the names of 657 different colors. To see them all, look at the output of the `colors()` function.

```
allcolors <- colors()
head(allcolors) # some colors

## [1] "white"          "aliceblue"      "antiquewhite"
## [4] "antiquewhite1" "antiquewhite2" "antiquewhite3"

length(allcolors) # number of color names

## [1] 657
```

However, R can produce many more colors than this. To refer to a color from the full range of possible colors, we can use the *hexadecimal color code. Hexadecimal* is a number system whose base is 16, with integers 0–9 and letters A–F representing values from 0 to 15. A hexadecimal color code is a sequence of six characters beginning with a hash sign (#). Each set of two digits represents the strength of the three primary colors—red, green, and blue, or *RGB*—with each taking a value from 0 to 255 (or one of 2^8 levels). For example, half-strength red and blue together yields purple. This can be represented as RGB = (127, 0, 127). Recognizing that 127 is equal to 7F in the base-16 numeral system, we arrive at the hexadecimal color code of #7F007F.

In R, the `rgb()` function helps create hexadecimal color codes from numerical values. The three arguments, `red`, `green`, and `blue`, take the intensity of each color, ranging from 0 to 1, which gets translated into an integer value from 0 to 255 and then represented as a hexadecimal numeral. In addition, we can create more than one color code from `rgb()` at a time. The arguments can take vectors of length longer than 1. Below are some examples of hexadecimal color code. There are also many online sources that help us find the hexadecimal representation of a color. We start with primary colors.

```
red <- rgb(red = 1, green = 0, blue = 0) # red
green <- rgb(red = 0, green = 1, blue = 0) # green
blue <- rgb(red = 0, green = 0, blue = 1) # blue
c(red, green, blue) # results

## [1] "#FF0000" "#00FF00" "#0000FF"
```

Black and white can be represented by 0% or 100% for each primary color, respectively.

```
black <- rgb(red = 0, green = 0, blue = 0) # black
white <- rgb(red = 1, green = 1, blue = 1) # white
c(black, white) # results

## [1] "#000000" "#FFFFFF"
```

Finally, we can create purple (50% red and 50% blue) and yellow (100% red and 100% green). The `rgb()` function can take a vector of inputs, as illustrated in this example.

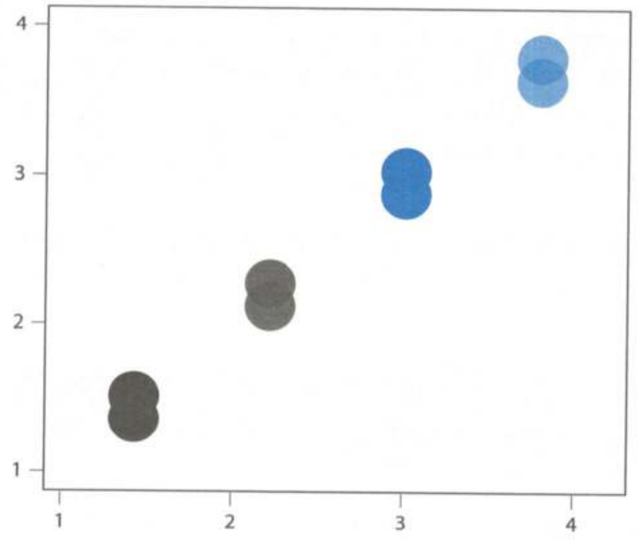
```
rgb(red = c(0.5, 1), green = c(0, 1), blue = c(0.5, 0))

## [1] "#800080" "#FFFF00"
```

Another advantage of using hexadecimal color codes is that we can make the colors (partly) transparent by adding two additional digits, from 00 to FF, to the end of a hexadecimal color code. This enables us to control the level of transparency. Again, it is easier to think about the intensity scale from 0 to 1 and use the `rgb()` function to transform it to a hexadecimal color code. The function takes a fourth argument `alpha`, which can be used for this purpose. An example is given here.

```
## semitransparent blue
blue.trans <- rgb(red = 0, green = 0, blue = 1, alpha = 0.5)
## semitransparent black
black.trans <- rgb(red = 0, green = 0, blue = 0, alpha = 0.5)
```

Once we know the hexadecimal colors, we can use them (as a character object) in our plots in the same way that we have been using named colors like `"red"` and `"blue"`. In the following plot, semitransparent circles can be easily distinguished even when they overlap, whereas it is harder to distinguish between nontransparent circles. Note that in this plot we suppress the default axis labels in order to avoid distraction by setting the `ann` argument to `FALSE` in the `plot()` function.

```
## completely colored dots; difficult to distinguish
plot(x = c(1, 1), y = c(1, 1.2), xlim = c(0.5, 4.5), ylim = c(0.5, 4.5),
     pch = 16, cex = 5, ann = FALSE, col = black)
points(x = c(3, 3), y = c(3, 3.2), pch = 16, cex = 5, col = blue)
## semitransparent; easy to distinguish
points(x = c(2, 2), y = c(2, 2.2), pch = 16, cex = 5, col = black.trans)
points(x = c(4, 4), y = c(4, 4.2), pch = 16, cex = 5, col = blue.trans)
```



5.3.4 US PRESIDENTIAL ELECTIONS

Now that we understand how color is represented in R, we can color maps. Here, we color the map of the United States using the 2008 presidential election results. The election data were introduced in chapter 4. The names and description of variables in the data file `pres08.csv` are given in table 4.1.

We will color each state in two ways. First, we use blue for the states won by Obama and red for the states won by McCain. This will produce a map just like figure 4.1 with "blue and red states." Second, we exploit the fact that various shades of purple can be produced as a mixture of blue and red in the RGB color scheme. Specifically, we compute the two-party vote share and set the intensity of blue as the Democratic two-party vote share and that of red as the Republican two-party vote share. In this way, the color of a state reflects the degree of support for Democratic and Republican candidates. The following code chunk loads the data set, computes the two-party vote shares, and sets the RGB color scheme for California based on its two-party vote share.

```
pres08 <- read.csv("pres08.csv")
## two-party vote share
pres08$Dem <- pres08$Obama / (pres08$Obama + pres08$McCain)
pres08$Rep <- pres08$McCain / (pres08$Obama + pres08$McCain)
## color for California
cal.color <- rgb(red = pres08$Rep[pres08$state == "CA"],
                 blue = pres08$Dem[pres08$state == "CA"],
                 green = 0)
```

We now color the map of California in two ways. First, we color it as a blue state because Obama won California in 2008. Second, we color it using the RGB color scheme based on the two-party vote share. To add color to a map, we must specify the `col` argument. In addition, we set the `fill` argument to `TRUE` in order to fill each state with the specified color.

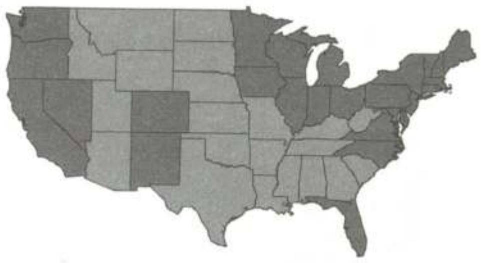
```
## California as a blue state
map(database = "state", regions = "California", col = "blue",
    fill = TRUE)
## California as a purple state
map(database = "state", regions = "California", col = cal.color,
    fill = TRUE)
```

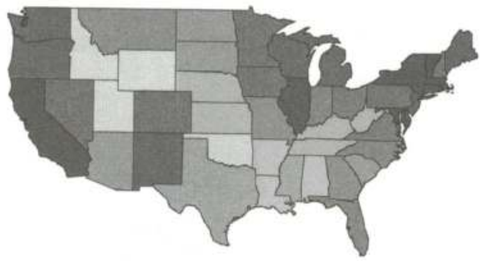
The right plot below uses gray instead of purple. See page C6 for the full-color version.

We will repeat this for all states using a loop. The map does not include Hawaii, Alaska, and Washington DC, so we will skip those states. Note that we will set the `add` argument to `TRUE` in order to add a color to each state. A loop is used because we color one state at a time. We first use a dichotomized color scheme where the states Obama won appear blue and those won by McCain are shown as red. In the second map, we use the RGB color scheme based on the two-party vote share for each state. The code chunks used for these two maps are almost identical. The only difference is the way in which color is chosen for each state.

```
## USA as red and blue states
map(database = "state") # create a map
for (i in 1:nrow(pres08)) {
    if ((pres08$state[i] != "HI") & (pres08$state[i] != "AK") &
        (pres08$state[i] != "DC")) {
        map(database = "state", regions = pres08$state.name[i],
          col = ifelse(pres08$Rep[i] > pres08$Dem[i], "red", "blue"),
            fill = TRUE, add = TRUE)
    }
}
## USA as purple states
map(database = "state") # create a map
for (i in 1:nrow(pres08)) {
    if ((pres08$state[i] != "HI") & (pres08$state[i] != "AK") &
        (pres08$state[i] != "DC")) {
        map(database = "state", regions = pres08$state.name[i],
            col = rgb(red = pres08$Rep[i], blue = pres08$Dem[i],
                green = 0), fill = TRUE, add = TRUE)
    }
}
```

The maps below use gray scale. See page C7 for the full-color version.





The left-hand map shows that Obama won many states on the West and East Coasts whereas McCain was particularly strong in the Midwest. However, the right-hand map illustrates that no state is completely dominated by either Democrats or Republicans. Each state has both types of voters, and it is the winner-take-all electoral system that is responsible for characterizing each state as either a blue or a red state.

Table 5.5. Walmart Store Opening Data.

Variable	*Description*
`opendate`	opening date for the store
`st.address`	street address of the store
`city`	city of the store
`state`	state of the store
`type`	store type (`Wal-MartStore`, `SuperCenter`, `DistributionCenter`)
`long`	longitude of the store
`lat`	latitude of the store

Note: The data set contains spatial and temporal information about Walmart store openings from the first opening on March 1, 1962 until August 1, 2006.

5.3.5 EXPANSION OF WALMART

Shifting from politics to business, we next examine the expansion of Walmart, a successful American multinational chain of retail discount department and warehouse stores.[9] Walmart opened its first store in 1962 in Rogers, Arkansas. Over the next several decades, it opened many stores within the United States and then around the world. Walmart has become one of the largest retail multinational companies in the world. Table 5.5 shows the names and descriptions of variables in the Walmart store opening data, `walmart.csv`. This data set contains spatial and temporal information about Walmart store openings, from the first opening on March 1, 1962 until August 1, 2006.

We begin by plotting all of the store locations on a map. The data set contains three different types of stores, represented by the variable `type`. `Wal-MartStore` represents a standard Walmart store, whereas `SuperCenter` is a standard Walmart store as well as a full supermarket. Walmart Supercenters often include pharmacies, garden shops, car service centers, and other specialty centers. We also plot `DistributionCenter` data, representing stores that distribute food and goods to standard Walmart stores and Supercenters. To distinguish the three types of stores, we use different colors—red for standard Walmart stores, green for Supercenters, and blue for Distribution Centers. We make the colors transparent so that circles representing different stores can overlap with each other. Distribution Centers, which are fewer than the other two types, will be represented by larger circles so that they stand out. The following code chunk defines these parameters.

```r
walmart <- read.csv("walmart.csv")
## red = Wal-MartStore, green = SuperCenter, blue = DistributionCenter
walmart$storecolors <- NA # create an empty vector
walmart$storecolors[walmart$type == "Wal-MartStore"] <-
    rgb(red = 1, green = 0, blue = 0, alpha = 1/3)
```

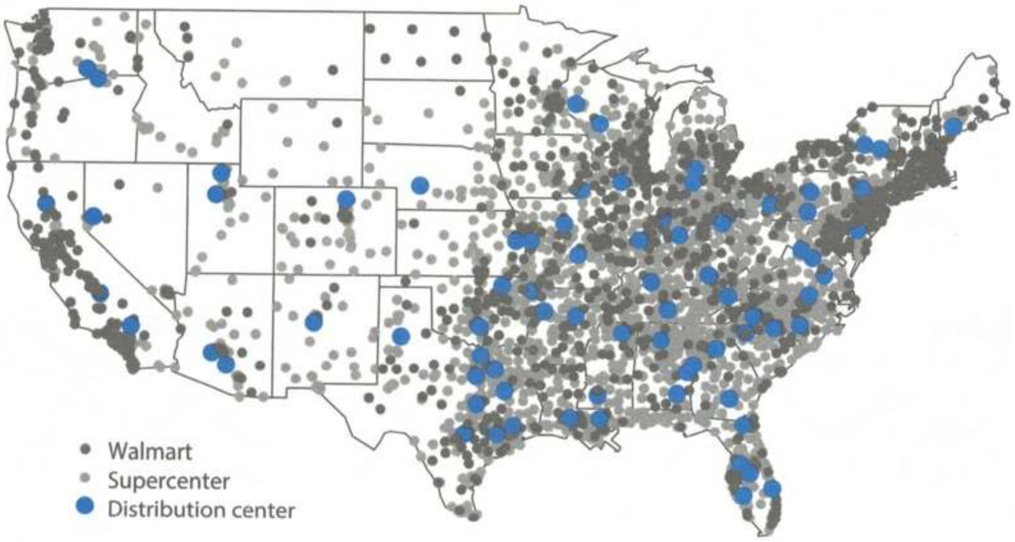
[9] This section is in part based on Thomas J. Holmes (2011) "The diffusion of Wal-Mart and economies of density." *Econometrica*, vol. 79, no. 1, pp. 253–302.

```
walmart$storecolors[walmart$type == "SuperCenter"] <-
    rgb(red = 0, green = 1, blue = 0, alpha = 1/3)
walmart$storecolors[walmart$type == "DistributionCenter"] <-
    rgb(red = 0, green = 0, blue = 1, alpha = 1/3)
## larger circles for DistributionCenter
walmart$storesize <- ifelse(walmart$type == "DistributionCenter", 1, 0.5)
```

Finally, we create a map and add Walmart store locations to it. We also include a legend using the `legend()` function. To use this function, we specify the location of the legend by setting the `x` and `y` coordinates and provide a vector of legend texts as the `legend` argument. A box encloses the legend by default when the `bty` argument is left unspecified, whereas setting the argument to `"n"` eliminates the box. As before, the `pch` argument can be used to specify types of objects to plot. We choose solid circles whose size can be controlled by the `pt.cex` argument.

```
## map with legend
map(database = "state")
points(walmart$long, walmart$lat, col = walmart$storecolors,
       pch = 19, cex = walmart$storesize)
legend(x = -120, y = 32, bty = "n",
       legend = c("Walmart", "Supercenter", "Distribution center"),
       col = c("red", "green", "blue"), pch = 19, # solid circles
       pt.cex = c(0.5, 0.5, 1)) # size of circles
```

The map below uses dark and light gray circles in place of red and green circles. See page C7 for the full-color version.



The map clearly shows the business strategy of Walmart. While Supercenters are widespread throughout the Midwest and South, they appear less prevalent in the Northeast and the West Coast, as well as in urban centers more generally. In these areas, Walmart has chosen not to expand past the standard discount store format.

5.3.6 ANIMATION IN R

The previous analysis of Walmart store openings ignored the temporal dimension even though the data set contains the opening date. By examining the spatial–temporal patterns rather than spatial patterns alone, we can better understand how Walmart expanded its stores over time. What visualization strategy should we employ to achieve this goal? We can create a series of maps over time, showing all stores at various points in time.

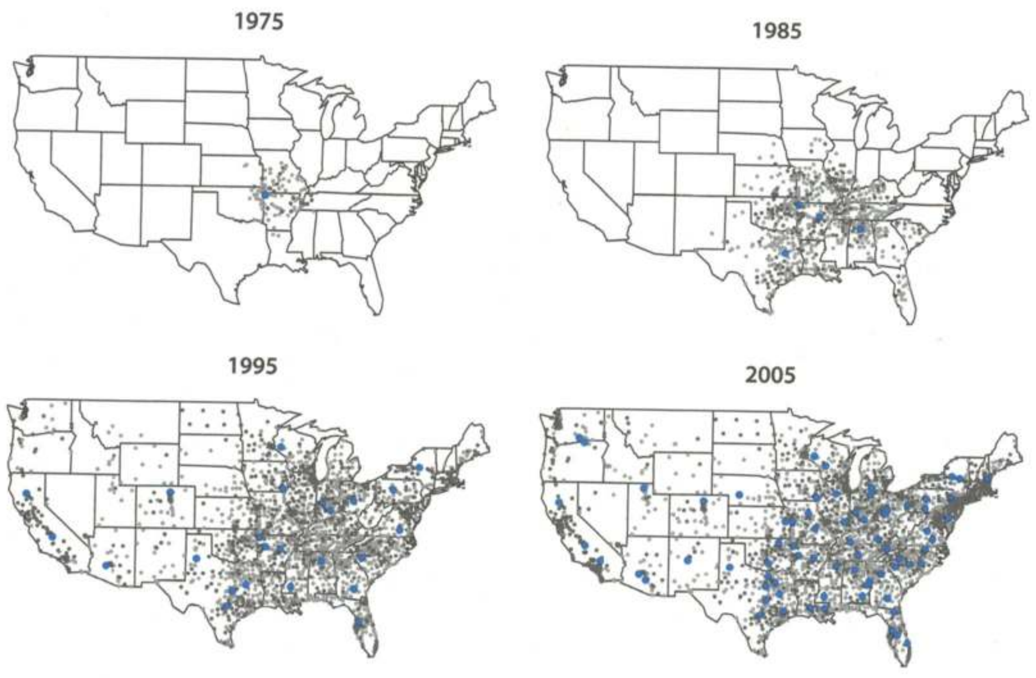
To do this, it is useful to define a function (see section 1.3.4) that subsets the data given a specified date, and then creates a map of Walmart stores like the one shown above. All we need to do is to include our previous code chunk in a function. Below, we create this function, called `walmart.map()`. The function takes two inputs. The first argument `data` takes a data frame, which should have a variable called `opendate` representing the opening date of the store. This variable should belong to the `Date` class. The second argument `date` takes another `Date` object defining the point in time for which the map should be created. The function subsets all the stores that opened on or before the specified date and then plots their locations on a map.

```
walmart.map <- function(data, date) {
    walmart <- subset(data, subset = (opendate <= date))
    map(database = "state")
    points(walmart$long, walmart$lat, col = walmart$storecolors,
           pch = 19, cex = walmart$storesize)
}
```

Using this function, it is straightforward to create a map at any given point of time. We create a map for every ten years.

```
walmart$opendate <- as.Date(walmart$opendate)
walmart.map(walmart, as.Date("1974-12-31"))
title("1975")
walmart.map(walmart, as.Date("1984-12-31"))
title("1985")
walmart.map(walmart, as.Date("1994-12-31"))
title("1995")
walmart.map(walmart, as.Date("2004-12-31"))
title("2005")
```

The following maps use dark and light gray circles in place of red and green circles. See page C8 for the full-color version.



Another method for visualizing spatial–temporal data like the above is *animation*, which dynamically shows how geographical patterns change over time. The **animation** package can show how Walmart has opened its stores at various locations at different times. We first set the number of maps to be animated and then create a vector of equally spaced dates from the beginning to the end of the data set.

```
n <- 25 # number of maps to animate
dates <- seq(from = min(walmart$opendate),
             to = max(walmart$opendate), length.out = n)
```

We are now ready to animate. At its core, using the **animation** package involves little more than writing a loop to create a series of maps over time. In fact, we need just one extra function, `saveHTML()`, to wrap the loop. The function takes the R code chunk as the main input, enclosed in curly braces `{ }`, and then inserts all plots that are created with the loop into an HTML file. The resulting HTML file can display the animation in a web browser. Useful arguments of the `saveHTML()` function include `htmlfile` for the HTML filename, `title` for the title of the animation, `outdir` for the name of the directory where the resulting files will be saved, and `autobrowse` indicating whether or not the output will be automatically displayed on a browser. In addition to HTML, available formats include `saveLatex()` for LaTeX files and `saveVideo()` for video files.

The following code chunk creates an animation and saves the HTML file named `walmart.html` and all other files to the working directory. Note that the `saveHTML()` function repeatedly calls the `walmart.map()` function we created earlier through a loop. The `getwd()` function returns the path to the working directory, and specifying this function as the `outdir` input will save all output files in that directory.

```
library("animation")
saveHTML({
    for (i in 1:length(dates)) {
        walmart.map(walmart, dates[i])
        title(dates[i])
    }
}, title = "Expansion of Walmart", htmlfile = "walmart.html",
        outdir = getwd(), autobrowse = FALSE)
```

We can play the animation by opening the resulting `walmart.html` file in a web browser. While watching, we see quite clearly the Southern origins of the Walmart franchise and its gradual spread throughout the region in the 1970s and 1980s. Particularly striking is the speed of the mid-1990s expansion throughout the rest of the country, as well as when and where new Distribution Centers are established in anticipation of regional expansion.

5.4 Summary

This chapter introduced types of data different from those we analyzed in the previous chapters. We focused on how to discover systematic patterns in a variety of data. We began by analyzing **textual data** under the **bag-of-words assumption** that ignores the sequence of words. By focusing on the frequency of different terms within and across documents, we can discover topics that underlie the corpus. We introduced **term frequency–inverse document frequency** as a statistic that measures the importance of each term in a particular document. Using *The Federalist Papers* as an example, we also showed how the frequency of words can predict the authorship of essays via a linear regression model. To assess prediction accuracy while avoiding overfitting, we used **cross validation** (and in particular a leave-one-out cross validation procedure).

The second type of data covered in this chapter was network data. We visualized both **directed** and **undirected network data** with **graphs**, where nodes (or vertices) represent units, and edges (or ties) between nodes represent the relationships between them. We showed how to compute various **centrality measures** in order to identify influential nodes within a given network. These measures include degree, closeness, and betweenness. We also introduced a popular iterative algorithm called **PageRank**, which forms the basis of the Google website ranking algorithm, as another way to measure centrality. These methods were illustrated through the classic example of the Florentine marriage network and a modern example of the Twitter-following network among politicians.

Finally, we considered **spatial** and **spatial–temporal data**. The spatial dimension splits into two types: spatial point and spatial polygon data. We showed how maps can visualize spatial patterns effectively using John Snow's famous study of a cholera outbreak in 19th century London. Snow utilized a natural experiment to uncover

Table 5.6. Constitution Preamble Data.

Variable	*Description*
`country`	country name with words separated by underscores
`year`	year the constitution was created
`preamble`	raw text of the preamble to the constitution

Note: The data set contains raw textual information about the preambles of constitutions around the world.

the primary cause of the outbreak. We also used maps to visualize the outcome of the US presidential election and the diffusion of Walmart stores over time. Like the analysis of texts and networks, visualization plays a central role in spatial data analysis. To investigate how spatial patterns change over time, we created an **animation** that sequentially displayed a series of maps. This visualization effectively demonstrated the expansion of Walmart stores in the United States over the last several decades.

5.5 Exercises

5.5.1 ANALYZING THE PREAMBLES OF CONSTITUTIONS

Some scholars argue that over the last few centuries, the US Constitution has emerged, either verbatim or paraphrased, in numerous founding documents across the globe.[10] Will this trend continue, and how might one even measure constitutional influence, anyway? One way is to see which constitutional rights (such as free speech) are shared across the founding documents of different countries, and observe how this commonality changes over time. An alternative approach, which we take in this exercise, is to examine textual similarity among constitutions. We focus on the preamble of each constitution, which typically states the guiding purpose and principles of the rest of the constitution. Table 5.6 presents the names and descriptions of the constitution preambles in `constitution.csv`.

1. First, let us visualize the data to better understand how constitutional documents differ. Start by importing the preamble data into a data frame, and then preprocess the text. Before preprocessing, use the `VectorSource()` function inside the `Corpus()` function. Create two data matrices for both the regular document-term frequency, and for the tf–idf weighted term frequency. In both cases, visualize the preamble to the US Constitution with a word cloud. How do the results differ between the two methods? Note that we must normalize the tf–idf weights by document size so that lengthy constitutions do not receive greater weights.

[10] This exercise is in part based on David S. Law and Mila Versteeg (2012) "The declining influence of the United States Constitution." *New York University Law Review*, vol. 87, no. 3, pp. 762–858 and Zachary Elkins, Tom Ginsburg, and James Melton (2012) "Comments on law and Versteeg's the declining influence of the United States Constitution." *New York University Law Review*, vol. 87, no. 6, pp. 2088–2101.

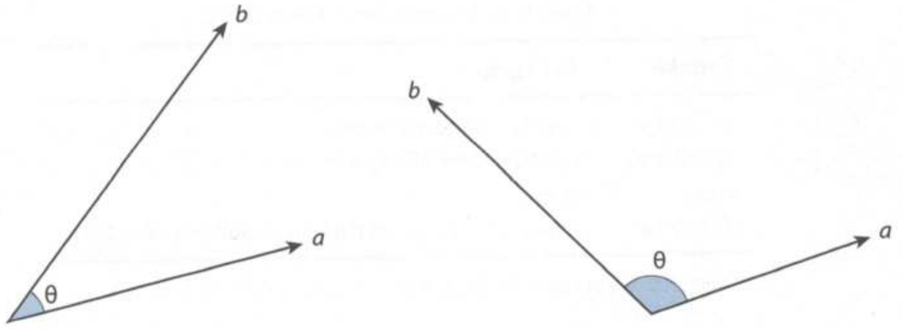


Figure 5.6. Cosine Similarity of Two Vectors. Two two-dimensional vectors a and b have a positive (negative) value of cosine similarity in the left (right) plot.

2. We next apply the k-means algorithm to the rows of the tf–idf matrix and identify clusters of similar constitution preambles. Set the number of clusters to 5 and describe the results. To make each row comparable, divide it by a constant such that each row represents a vector of unit length. Note that the length of a vector $a = (a_1, a_2, \ldots, a_n)$ is given by $||a|| = \sqrt{a_1^2 + a_2^2 + \cdots + a_n^2}$.

3. We will next see whether new foreign constitutions are more similar to the US Constitution preamble than the existing ones. In the document-term matrix, each document is represented as a vector of term frequencies. To compare two documents, we define *cosine similarity* as the cosine of the angle θ between the two corresponding n-dimensional vectors $a = (a_1, a_2, \ldots, a_n)$ and $b = (b_1, b_2, \ldots, b_n)$. Formally, the measure is defined as

$$\text{cosine similarity} = \cos\theta = \frac{a \cdot b}{||a|| \times ||b||} = \frac{\sum_{i=1}^n a_i b_i}{\sqrt{\sum_{i=1}^n a_i^2}\sqrt{\sum_{i=1}^n b_i^2}}.$$

The numerator represents the so-called *dot product* of a and b, while the denominator is the product of the lengths of the two vectors. The measure ranges from -1 (when the two vectors go in opposite directions) to 1 (when they completely overlap). As illustrated in figure 5.6, two vectors have a positive (negative) value of cosine similarity when they point in similar (different) directions. The measure is zero when they are perpendicular to each other.

Below is a function that takes a vector a, alongside a collection of vectors or a matrix b, and computes the cosine similarity between a and each row of b.

```
cosine <- function(a, b) {
    ## t() transposes a matrix ensuring that vector "a" is multiplied
    ## by each row of matrix "b"
    numer <- apply(a * t(b), 2, sum)
    denom <- sqrt(sum(a^2)) * sqrt(apply(b^2, 1, sum))
    return(numer / denom)
}
```

Table 5.7. International Trade Data.

Variable	*Description*
`country1`	country name of exporter
`country2`	country name of importer
`year`	year
`exports`	total value of exports (in tens of millions of dollars)

Note: The data are given for 1900, 1920, 1940, 1955, 1980, 2000, and 2009.

Apply this function to identify the 5 constitutions whose preambles most resemble that of the US Constitution.

4. We examine the influence of the US Constitution on other constitutions over time. We focus on the postwar period. Sort the constitutions chronologically and calculate, for every 10 years from 1960 until 2010, the average of cosine similarity between the US Constitution and the constitutions that were created during the previous decade. Plot the result. Each of these averages computed over time is called a *moving average*. Does similarity tend to increase, decrease, or remain the same over time? Comment on the pattern you observe.

5. We next construct directed, weighted network data based on the cosine similarity of constitutions. Specifically, create an adjacency matrix whose (i, j)th entry represents the cosine similarity between the ith and jth constitution preambles, where the ith constitution was created in the same year or after the jth constitution. This entry is zero if the ith constitution was created before the jth constitution. Apply the PageRank algorithm to this adjacency matrix. Briefly comment on the result.

5.5.2 INTERNATIONAL TRADE NETWORK

The size and structure of international trade flows vary significantly over time.[11] The volume of goods traded between countries has grown rapidly over the past century, as technological advances have lowered the cost of shipping and countries have adopted more liberal trade policies. At times, however, trade flows have decreased due to disruptive events such as major wars and the adoption of protectionist trade policies. In this exercise, we will explore some of these changes by examining the network of international trade over several time periods. The data file `trade.csv` contains the value of exports from one country to another in a given year. The names and descriptions of variables in this data set are given in table 5.7.

[11] This exercise is based in part on Luca De Benedictis and Lucia Tajoli (2011) "The world trade network." *The World Economy*, vol. 34, no. 8, pp. 1417–1454. The trade data are from Katherine Barbieri and Omar Keshk (2012) *Correlates of War Project Trade Data Set*, version 3.0. Available at `http://correlatesofwar.org`.

1. We begin by analyzing international trade as an unweighted, directed network. For every year in the data set, create an adjacency matrix whose entry (i, j) equals 1 if country i exports to country j. If this export is zero, then the entry equals 0. We assume that missing data, indicated by `NA`, represents zero trade. Plot the *network density*, which is defined over time as

$$\text{network density} = \frac{\text{number of edges}}{\text{number of potential edges}}.$$

The `graph.density()` function can compute this measure given an adjacency matrix. Interpret the result.

2. For the years 1900, 1955, and 2009, compute the measures of centrality based on degree, betweenness, and closeness (based on total degree) for each year. For each year, list the 5 countries that have the largest values of each of these centrality measures. How do the countries on the lists change over time? Briefly comment on the results.

3. We now analyze the international trade network as a weighted, directed network in which each edge has a nonnegative weight proportional to its corresponding trade volume. Create an adjacency matrix for such network data. For the years 1900, 1955, and 2009, compute the centrality measures from above for the weighted trade network. Instead of degree, however, compute the *graph strength*, which in this case equals the sum of imports and exports with all adjacent nodes. The `graph.strength()` function can be used to compute this weighted version of degree. For betweenness and closeness, we use the same function as before, i.e., `closeness()` and `betweenness()`, which can handle weighted graphs appropriately. Do the results differ from those of the unweighted network? Examine the top 5 countries. Can you think of another way to calculate centrality in this network that accounts for the value of exports from each country? Briefly discuss.

4. Apply the PageRank algorithm to the weighted trade network, separately for each year. For each year, identify the 5 most influential countries according to this algorithm. In addition, examine how the ranking of PageRank values has changed over time for each of the following 5 countries—United States, United Kingdom, Russia, Japan, and China. Briefly comment on the patterns you observe.

5.5.3 MAPPING US PRESIDENTIAL ELECTION RESULTS OVER TIME

The partisan identities of many states have been stable over time. For example, Massachusetts is a solidly "blue" state, having pledged its electoral votes to the Democratic candidate in 8 out of the last 10 presidential elections. On the other extreme, Arizona's electoral votes went to the Republican candidate in 9 of the same 10 elections. Still, geography can occasionally be a poor predictor of presidential elections.

Table 5.8. County-Level US Presidential Election Data.

Variable	*Description*
`state`	full name of the 48 states (excluding Alaska, Hawaii, and the District of Columbia)
`county`	county name
`year`	election year
`rep`	popular votes for the Republican candidate
`dem`	popular votes for the Democratic candidate
`other`	popular votes for other candidates

For instance, in 2008, typically red states—including North Carolina, Indiana, and Virginia—helped elect Barack Obama to the presidency.

In this exercise, we will again map the US presidential election results for 48 states. However, our data will be more detailed in two respects. First, we will analyze data from 14 presidential elections from 1960 to 2012, allowing us to visualize how the geography of party choice has changed over time. Second, we will examine election results at the county level, allowing us to explore the spatial distribution of Democratic and Republican voters within states. The data file is available in CSV format as `elections.csv`. Each row of the data set represents the distribution of votes in that year's presidential election from each county in the United States. Table 5.8 presents the names and descriptions of variables in this data set.

1. We begin by visualizing the outcome of the 2008 US presidential election at the county level. Begin with Massachusetts and Arizona and visualize the county-level outcome by coloring counties based on the two-party vote share as done in section 5.3.4. The color should range from pure blue (100% Democratic) to pure red (100% Republican) using the RGB color scheme. Use the `county` database in the **maps** package. The `regions` argument of the `map()` function enables us to specify the state and county. The argument accepts a character vector, each entry of which has the syntax `state, county`. Provide a brief comment.

2. Next, using a loop, visualize the 2008 county-level election results across the United States as a whole. Briefly comment on what you observe.

3. We now examine how the geographical distribution of US presidential election results has changed over time at the county level. Starting with the 1960 presidential election, which saw Democratic candidate John F. Kennedy prevail over Republican candidate Richard Nixon, use animation to visualize the county-level election returns for each presidential election up to 2012. Base your code on what you programmed to answer the previous question.

4. In this exercise, we quantify the degree of partisan segregation for each state. We consider a state to be politically segregated if Democrats and Republicans tend

Chapter 6

Probability

Probability is the very guide of life.
—Cicero, *De Natura*

Until now, we have studied how to identify patterns in data. While some patterns are indisputably clear, in many cases we must figure out ways to distinguish systematic patterns from noise. Noise, also known as random error, is the irrelevant variation that occurs in every real-world data set. Quantifying the degree of statistical uncertainty of empirical findings is the topic for the *next* chapter, but this requires an understanding of probability. Probability is a set of mathematical tools that measure and model randomness in the world. As such, this chapter introduces the derivation of the fundamental rules of probability, with the use of mathematical notation. In the social sciences, we use probability to model the randomly determined nature of various real-world events, and even human behavior and beliefs. Randomness does not necessarily imply complete unpredictability. Rather, our task is to identify systematic patterns from noisy data.

6.1 Probability

We use *probability* as a measure of uncertainty. Probability is based on a set of three simple axioms, from which a countless number of useful theorems have been derived. In this section, we show how to define, interpret, and compute probability.

6.1.1 FREQUENTIST VERSUS BAYESIAN

In everyday life, we often hear statements such as "the probability of winning a coin toss is 50%" and "the probability of Obama winning the 2008 US presidential election is 80%." What do we mean by "probability"? There are at least two different interpretations. One interpretation, which is called the *frequentist* interpretation, states that probability represents the *limit* of relative frequency, defined as the ratio between the number of times the event occurs and the number of trials, in repeated trials under the same conditions. For example, the above statement about coin tosses can be interpreted as follows: if a coin toss is repeatedly conducted under the same conditions, to live in different counties. A common way to quantify the degree of residential segregation is to use the *dissimilarity index* given by

$$\text{dissimilarity index} = \frac{1}{2}\sum_{i=1}^{N}\left|\frac{d_i}{D} - \frac{r_i}{R}\right|.$$

In the formula, d_i (r_i) is the number of Democratic (Republican) votes in the ith county and D (R) is the total number of Democratic (Republican) votes in the state. N represents the number of counties. This index measures the extent to which Democratic and Republican votes are evenly distributed within states. It can be interpreted as the percentage of one group that would need to move in order for its distribution to match that of the other group. Using data on Democratic and Republican votes from the 2008 presidential election, calculate the dissimilarity index for each state. Which states are among the most (least) segregated according to this measure? Visualize the result as a map. Briefly comment on what you observe.

5. Another way to compare political segregation across states is to assess whether counties within a state are highly unequal in terms of how many Democrats or Republicans they have. For example, a state would be considered segregated if it had many counties filled with Democrats and many with no Democrats at all. In chapter 3, we considered the Gini coefficient as a measure of inequality (see section 3.6.2). Calculate the Gini coefficient for each state based on the percentage of Democratic votes in each county. Give each county the same weight, disregarding its population size. Which states have the greatest (or lowest) value of the index? Visualize the result using a map. What is the correlation between the Gini index and the dissimilarity index you calculated above? How are the two measures conceptually and empirically different? Briefly comment on what you observe and explain the differences between the two indexes. To compute the Gini index, use the `ineq()` function in the **ineq** package by setting its argument `type` to `"Gini"`.

6. Lastly, we examine how the degree of political segregation has changed in each state over time. Use animation to visualize these changes. Briefly comment on the patterns you observe.

Figure 6.1. Reverend Thomas Bayes (1701–1761).

the fraction of times a coin lands on heads approaches 0.5 as the number of coin tosses increases. Here, the mathematical term, "limit," represents the value to which a sequence of relative frequencies converges as the number of (hypothetically) repeated experiments approaches infinity.

The frequentist interpretation of probability faces several difficulties. First, it is unclear what we mean by "the same conditions." In the case of coin flips, such conditions may include initial angle and velocity as well as air pressure and temperature. However, if all conditions are identical, then the laws of physics imply that a coin flip will always yield the same outcome. Second, in practice, we can never conduct experiments like coin flips under the exact same conditions infinitely many times. This means that probability may be unable to describe the randomness of many events in the real world. In fact, coin flips may be among the easiest experiments to repeat under nearly identical conditions. Many other events covered in this book happen in dynamic social environments that are constantly changing.

How should we think about the probability of Obama winning the 2008 US presidential election from the frequentist perspective? Since the 2008 US presidential election occurs only once, it is strange to consider a hypothetical scenario in which this particular election occurs repeatedly under the same conditions. In addition, since Obama either wins the election or not, the probability of his victory should be either 0 or 1. Here, what is random is the election forecast (due to sampling variability etc.) not the actual election outcome.

An alternative framework is the *Bayesian* interpretation of probability, named after an 18th century English mathematician and minister, Thomas Bayes (see figure 6.1). According to this paradigm, probability is a measure of one's subjective belief about the likelihood of an event occurring. A probability of 0 means that an individual thinks an event is impossible, whereas a probability of 1 implies that the individual thinks the event is sure to happen. Any probability value between 0 and 1 indicates the degree to which one feels uncertain about the occurrence of the event. In contrast to the frequentist perspective, the Bayesian framework makes it easy to interpret the statement, "the probability of Obama winning the 2008 US presidential election is x%," because x simply reflects the speaker's subjective belief about the likelihood of Obama's victory.

Critics of Bayesian interpretation argue that if scientists have identical sets of empirical evidence, they should arrive at the same conclusion rather than reporting different probabilities of the same event. Such subjectivity may hinder scientific progress because under the Bayesian framework, probability simply becomes a tool to describe one's belief system. In contrast, Bayesians contend that human beings, including scientists, are inherently subjective, so they should explicitly recognize the role of their subjective beliefs in scientific research.

Regardless of the ongoing controversy about its interpretation, probability was established as a mathematical theory by Soviet mathematician Andrey Kolmogorov in the early 20th century. Since both frequentists and Bayesians use this mathematical theory, the disagreement is about interpretation and is not mathematical.

There are two dominant ways to interpret probability. According to the **frequentist framework**, probability represents the limit of the relative frequency with which an event of interest occurs when the number of experiments repeatedly conducted under the same conditions approaches infinity. The **Bayesian framework**, in contrast, interprets probability as one's subjective belief about the likelihood of event occurrence.

6.1.2 DEFINITION AND AXIOMS

We define probability using the following three concepts: *experiment*, *sample space*, and *event*.

The definition of probability requires the following concepts:

1. **experiment**: an action or a set of actions that produce stochastic events of interest
2. **sample space**: a set of all possible outcomes of the experiment, typically denoted by Ω
3. **event**: a subset of the sample space

We can briefly illustrate each concept using the aforementioned two examples. Flipping a coin or holding an election would be the experiment, while the sample space would be given by {lands on heads, lands on tails} or {Obama wins, McCain wins, a third-party candidate wins}. The mathematical term *set* refers to a collection of distinct objects. An event represents *any* subset of sample space, and hence it may include multiple outcomes. In fact, the entire sample space that contains all outcomes is also an event. Moreover, an event is said to *occur* if the set that defines the event includes an actual outcome of the experiment. In the election example, events include {Obama wins, McCain wins}, which contains two outcomes and can be understood in English as "either Obama or McCain wins." Since Obama won the election, this event did occur in 2008.

As another example, consider a voter's decision in the 2008 US presidential election as an experiment. The idea is that a voter's decision can be modeled as a stochastic, rather than deterministic, event. By considering all four possible outcomes, we can define the sample space of this experiment as Ω = {abstain, vote for Obama, vote for McCain, vote for a third-party candidate}. Within this sample space, we may consider the occurrence of various events including {vote for Obama, vote for McCain, vote for a third-party candidate} (i.e., "do not abstain") and {abstain, vote for McCain, vote for a third-party candidate} (i.e., "do not vote for Obama").

We now discuss how to compute probability, starting with the simplest case in which all outcomes are equally likely to occur. In this case, the probability of event A occurring, denoted by $P(A)$, can be computed as the ratio of the number of elements in the corresponding set A to that in the entire sample space Ω:

$$P(A) = \frac{\text{number of elements in } A}{\text{number of elements in } \Omega}. \tag{6.1}$$

To illustrate this, consider an experiment of tossing a fair coin 3 times. In this experiment, if we denote {lands on heads} and {lands on tails} as H and T, respectively, then the sample space is equal to the set of 8 outcomes, $\Omega = \{HHH, HHT, HTH, HTT, THH, THT, TTH, TTT\}$. We can then compute the probability of, for example, landing on heads at least twice by counting the number of elements in the relevant set, $A = \{HHH, HHT, HTH, THH\}$. In this case, therefore, using the formula above we obtain $P(A) = 4/8 = 0.5$.

Having defined probability, we next consider its basic rules or *axioms*. Modern probability theory rests on the following three simple axioms. Remarkably, from these axioms, the entire theory of probability, including all the existing rules and theorems, can be derived.

The **probability axioms** are given by the following three rules:

1. The probability of any event A is nonnegative:

$$P(A) \geq 0.$$

3. The probability that one of the outcomes in the sample space occurs is 1:

$$P(\Omega) = 1.$$

3. (*Addition rule*) If events A and B are mutually exclusive, then

$$P(A \text{ or } B) = P(A) + P(B). \tag{6.2}$$

The first two axioms together imply that probability ranges from 0 to 1. To understand the last axiom, recall the previous example in which the 2008 US presidential

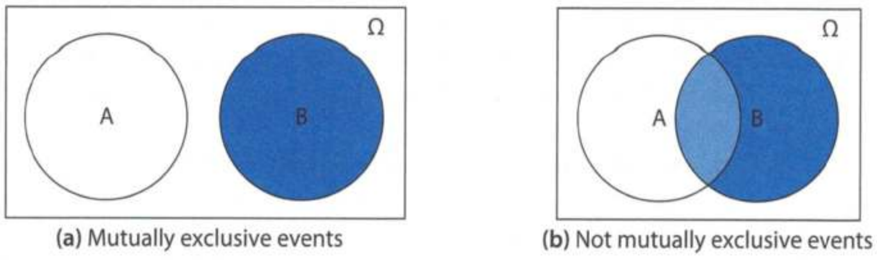

Ω
A
B
(a) Mutually exclusive events
Ω
A
B
(b) Not mutually exclusive events

Figure 6.2. Venn Diagram. Two events, A and B, can be mutually exclusive, having two disjoint sets of outcomes (left plot) or not mutually exclusive, sharing some outcomes (right plot). The rectangular box represents the sample space Ω. Source: Adapted from example by Uwe Ziegenhagen, http://texample.net.

election is considered as an experiment. "Mutually exclusive" in the last axiom means that two events, A and B, do not share an outcome. As illustrated by the *Venn diagram* (named after John Venn, an English philosopher) in figure 6.2a, mutually exclusive events imply two disjoint sets, meaning that they do not share any element. Consider two events: $A = \text{Obama wins}$ and $B = \text{McCain wins}$. Clearly, these two events are mutually exclusive in that both Obama and McCain cannot win at the same time. Hence, we can apply the addition rule to conclude that $P(\{\text{Obama wins}\} \text{ or } \{\text{McCain wins}\}) = P(\text{Obama wins}) + P(\text{McCain wins})$.

Now, consider two events that are not mutually exclusive because they share an outcome: $A = \text{Obama loses}$ and $B = \text{McCain loses}$. In this case, the addition rule does not apply because both A and B contain the same outcome: a third-party candidate wins. For events that are not mutually exclusive, we can apply the following general addition rule.

For any given events A and B, the **addition rule** is given by

$$P(A \text{ or } B) = P(A) + P(B) - P(A \text{ and } B). \tag{6.3}$$

Applying this to the current example, we have $P(\{\text{Obama loses}\} \text{or } \{\text{McCain loses}\}) = P(\text{Obama loses}) + P(\text{McCain loses}) - P(\{\text{Obama loses}\} \text{ and } \{\text{McCain loses}\})$.

This result can be immediately seen from the *Venn diagram* shown in figure 6.2b. In the diagram, we observe that the event, $\{A \text{ or } B\}$, can be decomposed into three mutually exclusive events, $\{A \text{ and } B^c\}$ (white region), $\{B \text{ and } A^c\}$ (dark blue region), and $\{A \text{ and } B\}$ (overlapped light blue region). The superscript c represents the *complement* of a set, which consists of all elements in the sample space except those in the set. For example, A^c represents the collection of all outcomes in the sample space that do not belong to A. The notation $\{A \text{ and } B^c\}$ translates to "all outcomes of A that do not belong to B." Since any outcome in the sample space belongs either to A or A^c, in general, we have

$$P(A^c) = 1 - P(A). \tag{6.4}$$

The equation directly follows from the probability axioms since events A and A^c are mutually exclusive and together they constitute the entire sample space.

Using the third probability axiom, given in equation (6.2), we have

$$P(A \text{ or } B) = P(A \text{ and } B^c) + P(B \text{ and } A^c) + P(A \text{ and } B). \tag{6.5}$$

When A and B are mutually exclusive, $P(A \text{ and } B^c)$ and $P(B \text{ and } A^c)$ reduce to $P(A)$ and $P(B)$, respectively (see figure 6.2a). In addition, we have $P(A \text{ and } B) = 0$ in this mutually exclusive case.

Finally, notice that event A can be decomposed as two mutually exclusive events, $\{A \text{ and } B\}$ (overlapped light blue region) and $\{A \text{ and } B^c\}$ (nonoverlapped white region). This is called the *law of total probability*.

For any given events A and B, the **law of total probability** is given by

$$P(A) = P(A \text{ and } B) + P(A \text{ and } B^c). \tag{6.6}$$

According to the law of total probability, we can write $P(A \text{ and } B^c) = P(A) - P(A \text{ and } B)$ by subtracting $P(A \text{ and } B)$ from both sides of equation (6.6). Similarly, the law of total probability can be applied to event B, yielding $P(B \text{ and } A^c) = P(B) - P(A \text{ and } B)$. Substituting these results into equation (6.5) and simplifying the expression leads to the general addition rule given in equation (6.3). We emphasize that this result is obtained by using the probability axioms alone. In addition, readers are encouraged to confirm the results shown in equations (6.3)–(6.6) using the Venn diagram of figure 6.2.

6.1.3 PERMUTATIONS

When each outcome is equally likely, in order to compute the probability of event A, we need to count the number of elements in event A as well as the total number of elements in the sample space Ω (see equation (6.1)). We introduce a useful counting technique, called *permutations*. Permutations refer to the number of ways in which objects can be arranged. For example, consider three unique objects A, B, and C. There are 6 unique ways to arrange them: $\{ABC, ACB, BAC, BCA, CAB, CBA\}$.

How can we compute the number of permutations without enumerating every arrangement, especially when the number of objects is large? It turns out that there is an easy way to do this. Let's consider the above example of arranging three objects, A, B, and C. First, there are three ways to choose the first object: A, B, or C. Once the first object is selected, there are two ways to choose the second object. Finally, the third object remains, leaving us only one way to choose this last object. We can conceptualize this process as a tree shown in figure 6.3, where the total number of leaves equals the number of permutations. Thus, to compute the number of leaves, we only need to sequentially multiply the number of branches at each level, i.e., $3 \times 2 \times 1$.

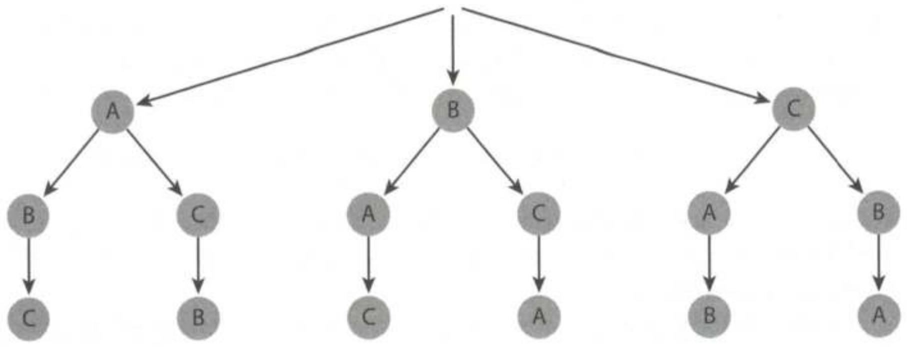


Figure 6.3. A Tree Diagram for Permutations. There are 6 ways to arrange 3 unique objects. Source: Adapted from example by Madit, http://texample.net.

Generalizing this idea, we can compute the number of permutations of k objects out of a set of n unique objects, denoted by ${}_nP_k$ where $k \leq n$, using the following formula.

The number of **permutations** when arranging k objects out of n unique objects is given by

$$ {}_nP_k = n \times (n-1) \times \cdots \times (n-k+2) \times (n-k+1) = \frac{n!}{(n-k)!}. \tag{6.7}$$

In this equation, ! represents the *factorial* operator. When n is a nonnegative integer, $n! = n \times (n-1) \times \cdots \times 2 \times 1$. Note that 0! is defined as 1.

In the previous example, $n=3$ and $k=3$. Therefore,

$$ {}_3P_3 = \frac{3!}{0!} = \frac{3 \times 2 \times 1}{1} = 6.$$

As another example, compute the number of ways in which you can arrange 4 cards out of 13 unique cards. This can be computed by setting $n=13$ and $k=4$ in equation (6.7):

$$ {}_{13}P_4 = \frac{13!}{(13-4)!} = 13 \times 12 \times 11 \times 10 = 17160.$$

The *birthday problem* is a well-known counterintuitive example of permutations. The problem asks how many people one needs in order for the probability that at least two people have the same birthday to exceed 0.5, assuming that each birthday is equally likely. What is surprising about this problem is that the answer is only 23 people, which is much lower than what most people guess. To solve this problem using permutations, first notice the following relationship:

$$\begin{aligned} &P(\text{at least two people have the same birthday}) \\ &\quad = 1 - P(\text{nobody has the same birthday}). \end{aligned} \tag{6.8}$$

This equality holds because the event {nobody has the same birthday} is the complement of the event {at least two people have the same birthday} (see equation (6.4)). This means that we only need to compute the probability that nobody has the same birthday.

Let k be the number of people. To compute the probability that nobody has the same birthday, we count the number of ways in which k people can have different birthdays. Since each birthday is assumed to be equally likely, we can use permutations to count the number of ways in which k unique birthdays can be arranged out of 365 days. This is given by ${}_{365}P_k = 365!/(365-k)!$. Applying equation (6.1), we then divide this number by the total number of elements in the sample space. The latter is equal to the total number of ways to select k possibly nonunique birthdays out of 365 days. The first person could have any of 365 days as his/her birthday, and so could any other person. Hence, the denominator is equal to $365 \times 365 \times \cdots \times 365 = 365^k$. Therefore, we have

$$\begin{aligned} &P(\text{nobody has the same birthday}) \\ &\quad = \frac{\text{\# of ways in which } k \text{ unique birthdays can be arranged}}{\text{\# of ways in which } k \text{ possibly nonunique birthdays can be arranged}} \\ &\quad = \frac{{}_{365}P_k}{365^k} = \frac{365!}{365^k(365-k)!}. \qquad (6.9) \end{aligned}$$

Together with equation (6.8), the solution to the birthday problem is $1 - 365!/\{365^k(365-k)!\}$.

Computing equation (6.9) is not easy even for a moderate value of k because both the denominator and numerator can take extremely large values. In such cases, it is often convenient to use the natural *logarithmic transformation* (see section 3.4.1). For the natural logarithm, $e^A = B$ implies $A = \log B$. In addition, the basic rules of logarithms we use here are

$$\log AB = \log A + \log B, \quad \log \frac{A}{B} = \log A - \log B, \quad \text{and} \quad \log A^B = B \log A.$$

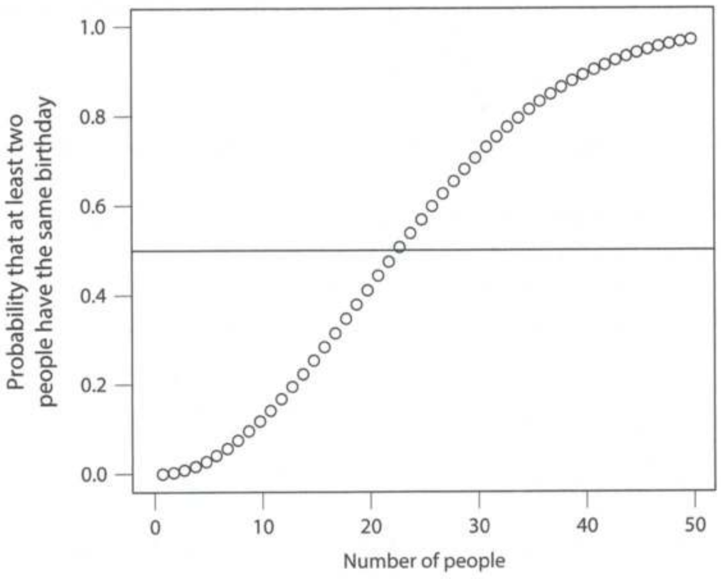
Applying these rules, we have

$$\log P(\text{nobody has the same birthday}) = \log 365! - k \log 365 - \log(365-k)!.$$

After computing this probability on a logarithmic scale, we then take the exponential transformation of it to obtain the desired probability. In R, we use the `lfactorial()` function to compute the logarithm of a factorial instead of the `factorial()` function, which computes a factorial without the logarithmic transformation. We now create a new function called `birthday`, which computes the probability that at least two people have the same birthday given k. The function is written so that it takes a vector of k values. We plot the results.

```
birthday <- function(k) {
    logdenom <- k * log(365) + lfactorial(365 - k) # log denominator
    lognumer <- lfactorial(365) # log numerator
    ## P(at least two have the same bday) = 1 - P(nobody has the same bday)
    pr <- 1 - exp(lognumer - logdenom) # transform back
    return(pr)
}
k <- 1:50
bday <- birthday(k)  # call the function
names(bday) <- k  # add labels
plot(k, bday, xlab = "Number of people", xlim = c(0, 50), ylim = c(0, 1),
     ylab = "Probability that at least two\n people have the same birthday")
abline(h = 0.5) # horizontal 0.5 line
bday[20:25]
```

```
##        20        21        22        23        24        25
## 0.4114384 0.4436883 0.4756953 0.5072972 0.5383443 0.5686997
```



We observe that when the number of people equals 23, the probability of at least two people having the same birthday exceeds 0.5. When the number of people is more than 50, this probability is close to 1.

6.1.4 SAMPLING WITH AND WITHOUT REPLACEMENT

While we derived an exact analytical solution to the birthday problem above, we can also produce an approximate solution using a *Monte Carlo simulation method*. The name originates from the Monte Carlo Casino in Monaco, but we may also simply call it a *simulation* method. The Monte Carlo simulation method refers to a general class of stochastic (as opposed to deterministic) methods that can be used to approximately solve analytical problems by randomly generating quantities of interest.

For the birthday problem, we sample k possibly nonunique birthdays out of 365 days and check whether or not the sampled k birthdays are all different. We use *sampling with replacement* because for each of k draws, every one of 365 days is equally likely to be sampled *regardless of* which dates were sampled before. In other words, the fact that one person is born on a certain day of the year should not exclude someone else from being born on the same day. After repeating this sampling procedure many times, we compute the fraction of simulation trials where at least two birthdays are the same, and this fraction serves as an estimate of the corresponding probability. This simulation procedure is intuitive because it emulates the *data-generating process*, or the actual process in which the data are generated, as described in the birthday problem.

In R, we can use the `sample()` function to implement sampling with or without replacement by setting the `replace` argument to either `TRUE` or `FALSE`. While unused in the birthday problem, *sampling without replacement* means that once an element is sampled, it will not be available for subsequent draws. For example, in the discussion of sample surveys in section 3.4.1, we introduced *simple random sampling* (SRS) as a method to randomly choose a representative sample of respondents from a population. SRS is an example of sampling without replacement because we typically do not interview the same individual multiple times. For sampling with replacement, the basic syntax is `sample(x, size = k, replace = TRUE)`, where `x` is a vector of elements to sample from, and `size` is the number of elements to choose. In addition, we can feed a vector of probability weights into the `prob` argument if unequal probabilities should be used to sample each element.

```r
k <- 23 # number of people
sims <- 1000 # number of simulations
event <- 0 # counter
for (i in 1:sims) {
    days <- sample(1:365, k, replace = TRUE)
    days.unique <- unique(days) # unique birthdays
    ## if there are duplicates, the number of unique birthdays
    ## will be less than the number of birthdays, which is "k"
    if (length(days.unique) < k) {
        event <- event + 1
    }
}
## fraction of trials where at least two bdays are the same
answer <- event / sims
answer

## [1] 0.509
```

While our simulation estimate is close to the analytical solution, which is 0.507, they are not identical. This difference is called the *Monte Carlo error*, but is the inevitable consequence of the simulation approach. The size of the Monte Carlo error depends on the nature of the problem and it differs from one simulation to another. It is difficult to eliminate such an error but it is possible to reduce it. To obtain a more accurate estimate, we increase the number of simulations. In the above code, we set the number of simulations to 1000. Next, we run the same code with the number of simulations set to one million and obtain an estimate of 0.508, which is closer to the true answer.

The **Monte Carlo simulation method** refers to a general class of repeated random sampling procedures used to approximately solve analytical problems. Commonly used procedures include **sampling with replacement**, in which the same unit can be repeatedly sampled, and **sampling without replacement**, in which each unit can be sampled at most once.

6.1.5 COMBINATIONS

We introduce another useful counting method called *combinations*. Combinations are similar to permutations, but the former ignores ordering while the latter does not. That is, combinations are ways to choose k distinct elements out of n elements without regard to their order. This means that when choosing 2 elements, two *different* permutations, AB and BA, represent one *identical* combination. Since the order in which the elements are arranged does not matter, the number of combinations is never greater than the number of permutations. For example, if we choose 2 distinct elements out of 3 elements, A, B, and C, the number of permutations is ${}_3P_2 = 6$ (AB, BA, AC, CA, BC, CB), whereas the number of combinations is 3 (AB, AC, BC).

In fact, to compute combinations, we first calculate permutations ${}_nP_k$ and then divide by $k!$. This is because given k sampled elements, there are $k!$ ways to arrange them in a different order, and yet all these arrangements are counted as a single combination. In the above example, for every set of two sampled elements (e.g., A and B), we have $2! (= 2 \times 1 = 2)$ ways of arranging them (i.e., AB and BA) but these two permutations count as one combination. Here, we obtain the number of combinations through the division of ${}_3P_2$ by $2!$. In general, the formula for combinations is given as follows.

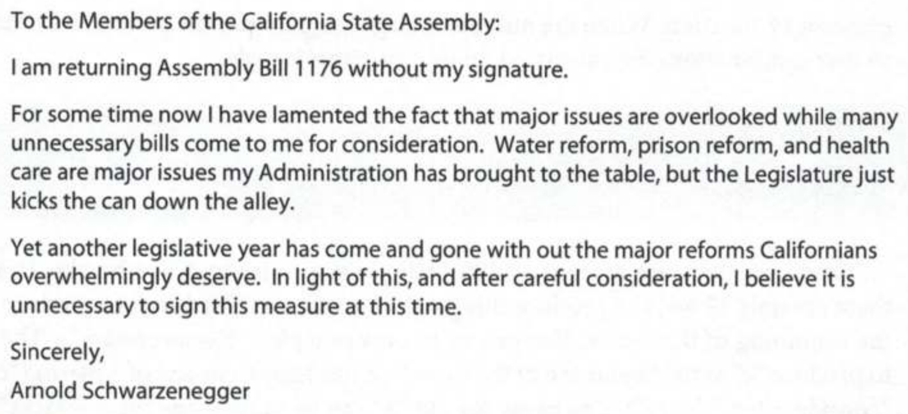
The number of **combinations** when choosing k distinct elements from n elements is denoted by either ${}_nC_k$ or $\binom{n}{k}$ and is computed as

$$ {}_nC_k = \binom{n}{k} = \frac{{}_nP_k}{k!} = \frac{n!}{k!(n-k)!}. \qquad (6.10) $$

Suppose that we are creating a committee of 5 out of 20 people (10 men and 10 women). Assume that each person is equally likely to be assigned to the committee. What is the probability that at least 2 women are on the committee? To compute this probability, we first note the following equality:

$$ \begin{aligned} & P(\text{at least 2 women are on the committee}) \\ & \quad = 1 - P(\text{no woman is on the committee}) \\ & \qquad - P(\text{exactly 1 woman is on the committee}). \end{aligned} $$

To compute the two probabilities on the right-hand side of this equation, we count the total number of ways we can assign 5 people to the committee out of 20 people regardless of their gender. This is given by ${}_{20}C_5 = 15{,}504$. Similarly, the number of



To the Members of the California State Assembly:

I am returning Assembly Bill 1176 without my signature.

For some time now I have lamented the fact that major issues are overlooked while many
unnecessary bills come to me for consideration. Water reform, prison reform, and health
care are major issues my Administration has brought to the table, but the Legislature just
kicks the can down the alley.

Yet another legislative year has come and gone with out the major reforms Californians
overwhelmingly deserve. In light of this, and after careful consideration, I believe it is
unnecessary to sign this measure at this time.

Sincerely,
Arnold Schwarzenegger

Figure 6.4. **California Governor Arnold Schwarzenegger's Veto Message in 2009.**

ways in which we can have no woman on the committee is given by ${}_{10}C_0 \times {}_{10}C_5 = 252$ because there is ${}_{10}C_0$ way to choose no woman and there are ${}_{10}C_5$ ways to choose 5 out of 10 men. Thus, the probability of having no woman is 0.016. The number of ways in which we can have exactly 1 woman on the committee is ${}_{10}C_1 \times {}_{10}C_4 = 2100$, giving a probability of 0.135. Altogether, the probability of having at least 2 women on the committee equals $0.85 \approx 1 - 252/15504 - 2100/15504$.

As a more complex example of combinations, we discuss an incident that occurred in 2009 when California Governor Arnold Schwarzenegger wrote a message to the state assembly regarding his veto of Assembly Bill 1176.[1] This message is displayed in figure 6.4. When the message was released, many observed that the first letters of each line in the main text, starting with "F" and ending with "u," constitute a sentence of profanity. Asked whether this was intentional, Schwarzenegger's spokesman replied, "My goodness. What a coincidence. I suppose when you do so many vetoes, something like this is bound to happen." Below, we consider the probability of this acrostic happening by chance.

For the sake of simplicity, suppose that the Governor gave his veto message to his secretary who then typed it in her computer but hit the return key at random. That is, the 85 words ("For" to "time") were divided by (random) line breaks into 7 lines, each with at least one word. We further assume that there are no broken words, every way of breaking the lines was equally likely, and the total number of lines is fixed at seven. Under this scenario, what is the probability of the coincidence happening?

To compute this probability using equation (6.1), we first consider the number of ways in which the 85 words can be divided into 7 lines. Note that to end up with 7 lines, 6 line breaks must be inserted. A line break may be inserted before the second word, before the third word, ..., or before the 85th word. There are thus 84 places into which 6 line breaks must be inserted. How many ways can we insert line breaks into 6 out of these 84 places? To compute this number, we use combinations rather than permutations because the order in which 6 line breaks are inserted does not matter. (Of course, the words in the acrostic must be ordered in a particular way to generate the profanity.) Therefore, the application of combinations leads to ${}_{84}C_6 = 84!/(6!78!)$ equally likely partitions. To compute combinations in R, we use the

[1] This section is based on Philip B. Stark (2009) "Null and vetoed: Chance coincidence?" *Chance*, vol. 23, no. 4, pp. 43–46. Although there are a total of 86 words from For to time, we follow the original article and use 85.

`choose()` function. When the number is large, we may use the `lchoose()` function so that combinations are calculated on the logarithmic scale.

```
choose(84, 6)

## [1] 406481544
```

Therefore, there are more than 400 million ways to insert 6 line breaks. However, there are only 12 ways to produce this particular acrostic. The break to produce "u" at the beginning of the second line can be in only one place ("unnecessary"). The break to produce "c" at the beginning of the third line can happen in any of 3 places ("come," "consideration," "care"). The break for the "k" can be in only one place ("kicks"). The break for the "y" can be in any of two places ("Yet," "year"). The break for the "o" can be in any of two places ("overwhelmingly," "of"). The break for the "u" can be in only one place ("unnecessary"). These scenarios lead to $12 = 1 \times 3 \times 1 \times 2 \times 2 \times 1$. Hence, the probability that this randomization scheme would produce the acrostic is $12/_{84}C_6$, or about one in 34 million. The analysis suggests that according to this probabilistic model, the "coincidence" is a highly unlikely event.

6.2 Conditional Probability

We next introduce conditional probability, which concerns how the probability of an event changes after we observe other events. Conditional probability follows the rules of probability described in section 6.1. The difference is that conditional probability enables us to take into account observed evidence.

6.2.1 CONDITIONAL, MARGINAL, AND JOINT PROBABILITIES

We begin by defining the conditional probability of event A occurring, given the information that event B has occurred. This conditional probability, denoted as $P(A \mid B)$, has the following definition.

The **conditional probability** of event A occurring given that event B occurred is defined as

$$P(A \mid B) = \frac{P(A \text{ and } B)}{P(B)}. \tag{6.11}$$

In this equation, $P(A \text{ and } B)$ is the **joint probability** of both events occurring, whereas $P(B)$ is the **marginal probability** of event B. By rearranging, we obtain the **multiplication rule**

$$P(A \text{ and } B) = P(A \mid B)P(B) = P(B \mid A)P(A). \tag{6.12}$$

Using this rule, we can derive an alternative form of the ***law of total probability*** introduced in equation (6.6):

$$P(A) = P(A \mid B)P(B) + P(A \mid B^c)P(B^c). \tag{6.13}$$

To see the importance of conditioning, consider two couples who are both expecting twins. One couple had an ultrasound exam, but the technician was able to determine only that one of the two was a boy. The other couple did not find out the genders of their twins until the delivery when they saw the first baby was a boy. What is the probability that both babies are boys? Is this probability different between the two couples? We begin by noting that there are four outcomes in the sample space. Denoting the baby gender by "G" for girl and "B" for boy, respectively, we can represent the sample space by $\Omega = \{GG, GB, BG, BB\}$. For example, GB means that the elder twin is a girl and the younger one is a boy.

Then, for the first couple, the probability of interest is

$$\begin{aligned} P(BB \mid \text{at least one is a boy}) &= \frac{P(BB \text{ and } \{\text{at least one is a boy}\})}{P(\text{at least one is a boy})} \\ &= \frac{P(BB \text{ and } \{BB \text{ or } BG \text{ or } GB\})}{P(BB \text{ or } BG \text{ or } GB)} \\ &= \frac{P(BB)}{P(BB \text{ or } BG \text{ or } GB)} = \frac{1/4}{3/4} = \frac{1}{3}. \end{aligned}$$

The third equality follows from the fact that event BB is a subset of event $\{\text{at least one is a boy}\}$, i.e., BB and $\{BB \text{ or } BG \text{ or } GB\} = BB$.

In contrast, for the second couple, we have

$$\begin{aligned} P(BB \mid \text{elder twin is a boy}) &= \frac{P(BB \text{ and } \{\text{the elder twin is a boy}\})}{P(\text{elder twin is a boy})} \\ &= \frac{P(BB \text{ and } \{BB \text{ or } BG\})}{P(BB \text{ or } BG)} \\ &= \frac{P(BB)}{P(BB \text{ or } BG)} = \frac{1/4}{1/2} = \frac{1}{2}. \end{aligned}$$

Therefore, this example illustrates that the information upon which we condition matters. Knowing that the first baby is a boy, as opposed to knowing that at least one is a boy, gives a different conditional probability of the same event.

Probability and conditional probability can also be used to describe the characteristics of a population. For example, if 10% of a population of voters are black, then we may write $P(\text{black}) = 0.1$. We can interpret this probability as stating that if we randomly sample a voter from this population there is a 10% chance this voter is black. Similarly, $P(\text{black} \mid \text{hispanic or black})$ represents the population proportion of blacks among minority (i.e., black and Hispanic) voters.

As an illustration, we will use a random sample of 10,000 registered voters from Florida contained in the CSV file `FLVoters.csv`. Table 6.1 shows the names and descriptions of variables in this sample list of registered voters. To begin, we load the data and remove those voters who contain a missing value using the `na.omit()` function.

Table 6.1. Florida Registered Voter List Sample.

Variable	*Description*
`surname`	surname
`county`	county ID of the voter's residence
`VTD`	voting district ID of the voter's residence
`age`	age
`gender`	gender: `m` = male and `f` = female
`race`	self-reported race

```
FLVoters <- read.csv("FLVoters.csv")
dim(FLVoters) # before removal of missing data

## [1] 10000     6

FLVoters <- na.omit(FLVoters)
dim(FLVoters) # after removal

## [1] 9113    6
```

For the sake of illustration, we will treat this sample of 9113 voters as a population of interest. To compute the *marginal probability* for each racial category, we can use the `table()` and `prop.table()` functions (see section 2.5.2) and calculate the proportion of voters who belong to each racial group in this population.

```
margin.race <- prop.table(table(FLVoters$race))
margin.race

##
##       asian       black    hispanic      native       other
## 0.019203336 0.131021617 0.130802151 0.003182267 0.034017338
##       white
## 0.681773291
```

The result shows, for example, that $P(\text{black}) = 0.13$ and $P(\text{white}) = 0.68$. Similarly, we can obtain the marginal probabilities of gender as follows.

```
margin.gender <- prop.table(table(FLVoters$gender))
margin.gender

##
##         f         m
## 0.5358279 0.4641721
```

Therefore, we have $P(\text{female}) = 0.54$ and $P(\text{male}) = 0.46$. Next, to compute the *conditional probability* of race given gender, we can look at the proportion of each racial group among female voters and among male voters, separately.

```
prop.table(table(FLVoters$race[FLVoters$gender == "f"]))
##
##       asian       black    hispanic      native       other
## 0.016997747 0.138849068 0.136391563 0.003481466 0.032357157
##       white
## 0.671922998
```

The result suggests, for example, $P(\text{black} \mid \text{female}) = 0.14$ and $P(\text{white} \mid \text{female}) = 0.67$. Lastly, the *joint probability* of race and gender can be computed by calculating the proportion of voters who belong to specific racial and gender groups.

```
joint.p <- prop.table(table(race = FLVoters$race, gender = FLVoters$gender))
joint.p
##           gender
## race                 f           m
##   asian    0.009107868 0.010095468
##   black    0.074399210 0.056622408
##   hispanic 0.073082410 0.057719741
##   native   0.001865467 0.001316800
##   other    0.017337869 0.016679469
##   white    0.360035115 0.321738176
```

This joint probability table gives, for example, $P(\text{black and female}) = 0.07$ and $P(\text{white and male}) = 0.32$. From this joint probability, we can compute the marginal and conditional probability. First, to obtain the marginal probability, we apply the *law of total probability* given in equation (6.6). For example, we can compute the probability of being a black voter by

$$P(\text{black}) = P(\text{black and female}) + P(\text{black and male}).$$

Thus, summing over columns for each row results in the marginal probability of race. This operation yields results identical to those obtained above.

```
rowSums(joint.p)
##       asian       black    hispanic      native       other
## 0.019203336 0.131021617 0.130802151 0.003182267 0.034017338
##       white
## 0.681773291
```

Similarly, we can obtain the marginal probability of gender from the joint probability table by summing over racial categories. Since we have a total of six racial categories, we will extend the law of total probability given in equation (6.6) to

$$P(A) = \sum_{i=1}^{N} P(A \text{ and } B_i), \tag{6.14}$$

where $B_1, \ldots, B_N$ is a set of mutually exclusive events which together cover the entire sample space. In the current setting, for example, since racial categories are mutually exclusive, we have

$$\begin{aligned} P(\text{female}) = \; & P(\text{female and asian}) + P(\text{female and black}) \\ & + P(\text{female and hispanic}) + P(\text{female and native}) \\ & + P(\text{female and other}) + P(\text{female and white}). \end{aligned}$$

Therefore, the marginal probability of gender is obtained by summing over rows for each column of the joint probability table.

```
colSums(joint.p)
##         f         m
## 0.5358279 0.4641721
```

Finally, the *conditional probability* can be obtained as the ratio of joint probability to the marginal probability (see equation (6.11)). For example, the conditional probability of being black among female voters is calculated as

$$P(\text{black} \mid \text{female}) = \frac{P(\text{black and female})}{P(\text{female})} \approx \frac{0.074}{0.536} \approx 0.139,$$

which, as expected, is equal to what we computed earlier.

The results of this example are summarized in table 6.2. From the joint probability, both marginal and conditional probabilities can be obtained. To compute marginal probability, we sum over either rows or columns. Once marginal probability is obtained in this way, we can divide joint probability by marginal probability in order to calculate the desired conditional probability.

We can extend the definition of conditional probability to settings with more than two types of events. For events A, B, and C, the joint probability is defined as $P(A \text{ and } B \text{ and } C)$, whereas there are three marginal probabilities $P(A)$, $P(B)$, and $P(C)$. In this case, there are two types of conditional probabilities: the joint probability of two events conditional on the remaining event (e.g., $P(A \text{ and } B \mid C)$) and the

Table 6.2. An Example of a Joint Probability Table.

Racial groups	*Gender*		
	Female	*Male*	*Marginal prob.*
Asian	0.009	0.010	0.019
Black	0.074	0.057	0.131
Hispanic	0.073	0.058	0.131
Native	0.002	0.001	0.003
White	0.360	0.322	0.682
Other	0.017	0.017	0.034
Marginal prob.	0.536	0.464	1

Note: The table is based on Florida voter registration data. The marginal probability of gender (far right column) and that of race (bottom row) can be obtained by summing the joint probabilities over columns and over rows, respectively.

conditional probability of one event given the other two (e.g., $P(A \mid B \text{ and } C)$). These conditional probabilities can be defined analogously to the two-event case as

$$P(A \text{ and } B \mid C) = \frac{P(A \text{ and } B \text{ and } C)}{P(C)}, \tag{6.15}$$

$$P(A \mid B \text{ and } C) = \frac{P(A \text{ and } B \text{ and } C)}{P(B \text{ and } C)} = \frac{P(A \text{ and } B \mid C)}{P(B \mid C)}. \tag{6.16}$$

The second equality in equation (6.16) follows from the equality $P(A \text{ and } B \text{ and } C) = P(A \text{ and } B \mid C)P(C)$, which is obtained by rearranging the terms in equation (6.15).

To illustrate the above conditional probabilities, we create a new `age.group` variable indicating four age groups: 20 and below, 21–40, 41–60, and above 60.

```
FLVoters$age.group <- NA # initialize a variable
FLVoters$age.group[FLVoters$age <= 20] <- 1
FLVoters$age.group[FLVoters$age > 20 & FLVoters$age <= 40] <- 2
FLVoters$age.group[FLVoters$age > 40 & FLVoters$age <= 60] <- 3
FLVoters$age.group[FLVoters$age > 60] <- 4
```

The joint probability of age group, race, and gender can be calculated as a three-way table. Below, this three-way table is displayed as two separate two-way tables: one two-way (race and age group) table for female voters and the other two-way table for male voters.

```
joint3 <-
    prop.table(table(race = FLVoters$race, age.group = FLVoters$age.group,
                     gender = FLVoters$gender))
```

```
joint3
## , , gender = f
##
##           age.group
## race                  1            2            3
##   asian    0.0001097333 0.0026336004 0.0041698672
##   black    0.0016460002 0.0280917371 0.0257873368
##   hispanic 0.0015362669 0.0260068035 0.0273236036
##   native   0.0001097333 0.0004389334 0.0006584001
##   other    0.0003292000 0.0062548008 0.0058158674
##   white    0.0059256008 0.0796664106 0.1260836168
##           age.group
## race                  4
##   asian    0.0021946670
##   black    0.0188741358
##   hispanic 0.0182157358
##   native   0.0006584001
##   other    0.0049380007
##   white    0.1483594864
##
## , , gender = m
##
##           age.group
## race                  1            2            3
##   asian    0.0002194667 0.0028530670 0.0051574674
##   black    0.0016460002 0.0228245364 0.0189838692
##   hispanic 0.0016460002 0.0197520026 0.0221661363
##   native   0.0000000000 0.0004389334 0.0003292000
##   other    0.0004389334 0.0069132009 0.0055964007
##   white    0.0040601339 0.0750576100 0.1184022825
##           age.group
## race                  4
##   asian    0.0018654669
##   black    0.0131680018
##   hispanic 0.0141556019
##   native   0.0005486667
##   other    0.0037309338
##   white    0.1242181499
```

For example, the proportion of black female voters who are above 60 or P(black and above 60 and female) is equal to 0.019. Suppose that we wish to obtain the conditional probability of being black and female given that a voter is above 60 years old or P(black and female | above 60). Using equation (6.15), we can compute this conditional probability by dividing the joint probability by the marginal probability of being above 60 or P(above 60). To extract a specific joint probability from the above three-way table, we specify the corresponding value for each demographic characteristic.

```
## marginal probabilities for age groups
margin.age <- prop.table(table(FLVoters$age.group))
margin.age

##
##          1          2          3          4
## 0.01766707 0.27093164 0.36047405 0.35092725

## P(black and female | above 60)
joint3["black", 4, "f"] / margin.age[4]

##          4
## 0.05378361
```

According to equation (6.16), the conditional probability of being black given that a voter is female and above 60 years old or P(black | female and above 60) can be computed by dividing the three-way joint probability P(black and above 60 and female) by the two-way joint probability P(above 60 and female). To obtain this two-way joint probability, we can create a two-way joint probability table for age group and gender.

```
## two-way joint probability table for age group and gender
joint2 <- prop.table(table(age.group = FLVoters$age.group,
                           gender = FLVoters$gender))
joint2

##          gender
## age.group           f           m
##         1 0.009656535 0.008010534
##         2 0.143092286 0.127839350
##         3 0.189838692 0.170635356
##         4 0.193240426 0.157686821

joint2[4, "f"] # P(above 60 and female)

## [1] 0.1932404

## P(black | female and above 60)
joint3["black", 4, "f"] / joint2[4, "f"]

## [1] 0.09767178
```

6.2.2 INDEPENDENCE

Having defined conditional probability, we can now formally discuss the concept of *independence*. Intuitively, the independence of two events implies that the knowledge of one event does not give us any additional information about the occurrence of the other event. That is, if events A and B are independent of each other, the conditional probability of A given B does not differ from the marginal probability of A. Similarly, the conditional probability of B given A does not depend on A:

$$P(A \mid B) = P(A) \quad \text{and} \quad P(B \mid A) = P(B). \tag{6.17}$$

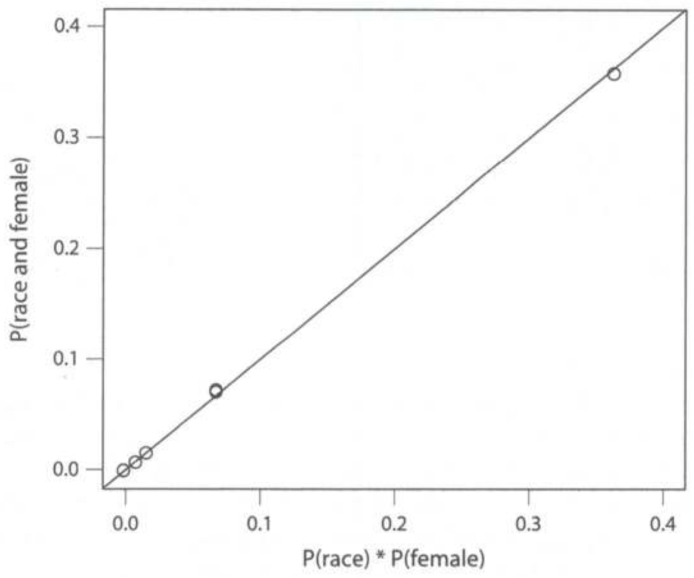
Together with equation (6.12), this equality implies the following formal definition of independence between events A and B.

Events A and B are **independent** if and only if the joint probability is equal to the product of the marginal probabilities:

$$P(A \text{ and } B) = P(A)P(B). \tag{6.18}$$

We investigate whether race and gender are independent of each other in the sample of Florida registered voters analyzed earlier. Although we do not expect this relationship to be exactly independent, we examine whether the proportion of female voters, for example, is greater than expected in some racial groups. Note that if independence holds, we should have, for example, $P(\text{black and female}) = P(\text{black})P(\text{female})$, $P(\text{white and male}) = P(\text{white})P(\text{male})$, and so on. We compare the products of marginal probabilities for race and female with their joint probabilities using a scatter plot. We use the `c()` function, which combines its inputs into a vector, to coerce a table format into a vector so that its elements can be used in the `plot()` function.

```
plot(c(margin.race * margin.gender["f"]), # product of marginal probs.
     c(joint.p[, "f"]), # joint probabilities
     xlim = c(0, 0.4), ylim = c(0, 0.4),
     xlab = "P(race) * P(female)", ylab = "P(race and female)")
abline(0, 1) # 45-degree line
```



The scatter plot shows that the points fall neatly along the 45-degree line, implying that $P(\text{race})P(\text{female})$ (horizontal axis) and $P(\text{race and female})$ (vertical axis) are approximately equal. This means that race and gender are approximately independent in this sample of registered voters. That is, the knowledge of a voter's gender does not help us predict her race. Similarly, one's race does not predict gender either.

The notion of independence extends to situations with more than two events. For example, if we have three events A, B, and C, the *joint independence* among these events implies that the joint probability can be written as the product of marginal probabilities:

$$P(A \text{ and } B \text{ and } C) = P(A)P(B)P(C). \tag{6.19}$$

Furthermore, we can define the independence between two events conditional on another event. The *conditional independence* of events A and B given event C implies that the joint probability of A and B given C is equal to the product of two conditional probabilities:

$$P(A \text{ and } B \mid C) = P(A \mid C)P(B \mid C). \tag{6.20}$$

Joint independence given in equation (6.19) implies pairwise independence given in equation (6.18). This result can be obtained by applying the *law of total probability*:

$$\begin{aligned} P(A \text{ and } B) &= P(A \text{ and } B \text{ and } C) + P(A \text{ and } B \text{ and } C^c) \\ &= P(A)P(B)P(C) + P(A)P(B)P(C^c) \\ &= P(A)P(B)\big(P(C) + P(C^c)\big) = P(A)P(B). \end{aligned}$$

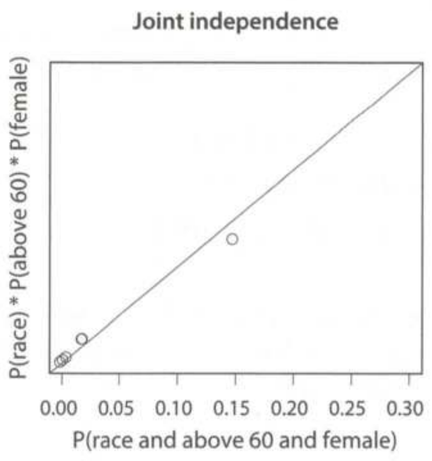
In addition, joint independence implies conditional independence, defined in equation (6.20), but the converse is not necessarily true. This result is based on the definition of conditional probability given in equation (6.15):

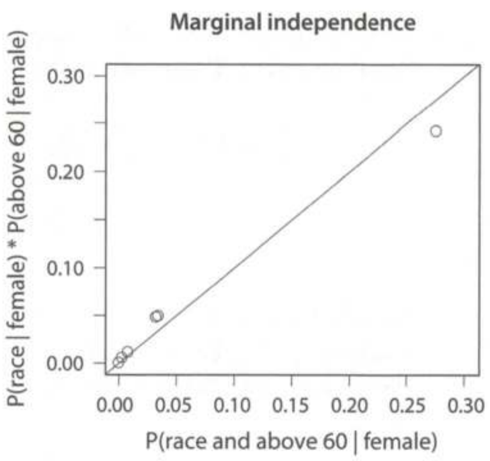
$$P(A \text{ and } B \mid C) = \frac{P(A \text{ and } B \text{ and } C)}{P(C)} = \frac{P(A)P(B)P(C)}{P(C)} = P(A \mid C)P(B \mid C).$$

The last equality follows from the fact that joint independence implies pairwise independence (and hence equation (6.17) holds for A and C as well as B and C).

To examine joint independence among our sample of registered Florida voters, we compare the elements of the three-way proportion table `joint3` with the corresponding product of marginal probabilities, `margin.race`, `margin.age`, and `margin.gender`. As an illustration, we set the age group to the above 60 category and examine female voters. We also examine conditional independence between race and gender, given age. For this, we again set the age and gender groups to the above 60 and female categories, respectively. The results show that both joint (left-hand plot) and conditional (right-hand plot) independence relationships approximately hold, despite small deviations.

```
## joint independence
plot(c(joint3[, 4, "f"]), # joint probability
     margin.race * margin.age[4] * margin.gender["f"], # product of marginals
     xlim = c(0, 0.3), ylim = c(0, 0.3), main = "Joint independence",
     xlab = "P(race and above 60 and female)",
     ylab = "P(race) * P(above 60) * P(female)")
abline(0, 1)
## conditional independence given female
plot(c(joint3[, 4, "f"]) / margin.gender["f"], # joint prob. given female
     ## product of marginals
     (joint.p[, "f"] / margin.gender["f"]) *
         (joint2[4, "f"] / margin.gender["f"]),
     xlim = c(0, 0.3), ylim = c(0, 0.3), main = "Marginal independence",
     xlab = "P(race and above 60 | female)",
     ylab = "P(race | female) * P(above 60 | female)")
abline(0, 1)
```





Finally, the well-known *Monty Hall problem* illustrates how tricky conditional probability and independence can be. The problem goes as follows. You are on a game show and must choose one of three doors, where one conceals a new car and two conceal old goats. After you randomly choose one door, the host of the game show, Monty, opens a different door, which does not conceal a car. Then, Monty asks you if you would like to switch to the (unopened) third door. You will win the new car if it is behind the door of your final choice. Should you switch, or stay with your original choice? Does switching make a difference? Most people think switching makes no difference because after Monty reveals one door with a goat, the two remaining doors have a goat or a car behind them. Therefore, the chance of winning a car is 50%. However, it turns out that this seemingly sensible reasoning is incorrect.

Let's think about this problem carefully. Consider the strategy of not switching. In this case, your initial choice determines the outcome regardless of what Monty does.

Therefore, the probability of winning the car is 1/3. Now, consider the strategy of switching. There are two scenarios. First, suppose that you initially choose a door with the car. The probability of this event is 1/3. Swapping the door in this scenario is a bad choice because you will not win the car. Next, suppose that the door you selected first has a goat. The probability of your initially choosing a door with a goat is 2/3. Then, since Monty opens another door with a goat, the remaining door to which you will switch contains a car. Hence, under this scenario, you will always win the car. Therefore, switching gives you a probability of winning the car that is twice as high as not switching.

We formalize this logic by applying the rules of probability covered so far. To compute the probability of winning a car given that you switch, we first apply the law of total probability in equation (6.13):

$$\begin{aligned} P(\mathsf{car}) &= P(\mathsf{car} \mid \mathsf{car\ first})P(\mathsf{car\ first}) + P(\mathsf{car} \mid \mathsf{goat\ first})P(\mathsf{goat\ first}) \\ &= P(\mathsf{goat\ first}) = \frac{2}{3}. \end{aligned}$$

To see why the second equality holds, notice that if you initially select the door with a car then switching makes you lose the car, i.e., $P(\mathsf{car} \mid \mathsf{car\ first}) = 0$. In contrast, if you first pick a door with a goat, then you have a 100% chance of winning a car by switching, i.e., $P(\mathsf{car} \mid \mathsf{goat\ first}) = 1$.

This rather counterintuitive problem can also be solved with *Monte Carlo simulations*. For emulating random choice in R, we use the `sample()` function. We set the `size` argument to `1` in order to randomly choose one element from a vector.

```
sims <- 1000
doors <- c("goat", "goat", "car")
result.switch <- result.noswitch <- rep(NA, sims)

for (i in 1:sims) {
    ## randomly choose the initial door
    first <- sample(1:3, size = 1)
    result.noswitch[i] <- doors[first]
    remain <- doors[-first] # remaining two doors
    ## Monty chooses one door with a goat
    if (doors[first] == "car") # two goats left
        monty <- sample(1:2, size = 1)
    else # one goat and one car left
        monty <- (1:2)[remain == "goat"]
    result.switch[i] <- remain[-monty]
}
mean(result.noswitch == "car")

## [1] 0.317

mean(result.switch == "car")

## [1] 0.683
```

6.2.3 BAYES' RULE

We discussed different interpretations of probability at the beginning of this chapter. One interpretation, proposed by Reverend Thomas Bayes, was that probability measures one's subjective belief in an event's occurrence. From this Bayesian perspective, it is natural to ask the question of how we should update our beliefs after observing some data. *Bayes' rule* shows how updating beliefs can be done in a mathematically coherent manner.

Bayes' rule is given by

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)} = \frac{P(B \mid A)P(A)}{P(B \mid A)P(A) + P(B \mid A^c)P(A^c)}. \tag{6.21}$$

In this equation, $P(A)$ is called the **prior probability** and reflects one's initial belief about the likelihood of event A occurring. After observing the data, represented as event B, we update our belief and obtain $P(A \mid B)$, which is called the **posterior probability**.

Regardless of whether we interpret probability as subjective belief, Bayes' rule shows mathematically how the knowledge of $P(A)$ (*prior probability*), $P(B \mid A)$, and $P(B \mid A^c)$ yields that of $P(A \mid B)$ (*posterior probability*). Bayes' rule is simply the result of rewriting the definition of conditional probability given in equation (6.11) using the law of total probability shown in equation (6.13):

$$P(A \mid B) = \frac{P(A \text{ and } B)}{P(B)} = \frac{P(B \mid A)P(A)}{P(B)}.$$

A well-known application of Bayes' rule is the interpretation of medical diagnostic tests, which can have false positives and false negatives (defined in section 4.1.3). Consider the following first-trimester screening test problem. A 35-year-old pregnant woman is told that 1 in 378 women of her age will have a baby with Down syndrome (DS). A first-trimester ultrasound screening procedure indicates that she is in a high-risk category. Of 100 cases of DS, 86 mothers will receive a high-risk result and 14 cases of DS will be missed. Also, there is a 1 in 20 chance for a normal pregnancy to be diagnosed as high risk. Given the result of the screening procedure, what is the probability that her baby has DS? What would the probability be if the result had been negative?

To solve this problem, we first specify the prior probability. Without any testing, the probability that a baby has DS, $P(\text{DS})$, is equal to 1/378 or approximately 0.003. The ultrasound screening procedure gives a high-risk result 86% of times when a baby actually has DS. This is called the *true positive rate* of the test and can be expressed as $P(\text{HR} \mid \text{DS}) = 0.86$, where HR denotes a high-risk result. However, the screening procedure also produces a *false positive rate* of 5%, which can be formally written as $P(\text{HR} \mid \text{not DS}) = 0.05$. Using this information, we can apply Bayes' rule to obtain the posterior probability that the baby has DS, given that the woman received a high-risk result, or the *positive predictive value* of the test:

$$P(\text{DS} \mid \text{HR}) = \frac{P(\text{HR} \mid \text{DS})P(\text{DS})}{P(\text{HR} \mid \text{DS})P(\text{DS}) + P(\text{HR} \mid \text{not DS})P(\text{not DS})}$$

$$= \frac{0.86 \times \frac{1}{378}}{0.86 \times \frac{1}{378} + 0.05 \times \frac{377}{378}} \approx 0.04.$$

Similarly, if the woman received a normal pregnancy result, the posterior probability becomes

$$P(\text{DS} \mid \text{not HR}) = \frac{P(\text{not HR} \mid \text{DS})P(\text{DS})}{P(\text{not HR} \mid \text{DS})P(\text{DS}) + P(\text{not HR} \mid \text{not DS})P(\text{not DS})}$$

$$= \frac{0.14 \times \frac{1}{378}}{0.14 \times \frac{1}{378} + 0.95 \times \frac{377}{378}} \approx 0.0004.$$

We see that even when the woman receives a high-risk result, the posterior probability of having a baby with DS is small. This is because DS is a relatively rare disease, as reflected by a small prior probability. As expected, if the woman receives a normal pregnancy result, then the posterior probability becomes even smaller than the prior probability.

We can use Bayes' rule to solve the Monty Hall problem introduced in section 6.2.2. Let A represent the event that the first door has a car behind it. Define B and C similarly for the second and third doors, respectively. Since each door is equally likely to have a car behind it, the prior probabilities are $P(A) = P(B) = P(C) = 1/3$. Suppose that we choose the first door and let MC represent the event that Monty opens the third door. We want to know whether switching to the second door increases the chance of winning the car, i.e., $P(B \mid \text{MC}) > P(A \mid \text{MC})$. We apply Bayes' rule after noting that $P(\text{MC} \mid A) = 1/2$ (Monty chooses between the second and third door with equal probability), $P(\text{MC} \mid B) = 1$ (Monty has no option but to open the third door, which has a goat), and $P(\text{MC} \mid C) = 0$ (Monty cannot open the third door, which has a car):

$$P(A \mid \text{MC}) = \frac{P(\text{MC} \mid A)P(A)}{P(\text{MC} \mid A)P(A) + P(\text{MC} \mid B)P(B) + P(\text{MC} \mid C)P(C)}$$

$$= \frac{\frac{1}{2} \times \frac{1}{3}}{\frac{1}{2} \times \frac{1}{3} + 1 \times \frac{1}{3} + 0 \times \frac{1}{3}} = \frac{1}{3},$$

$$P(B \mid \text{MC}) = \frac{P(\text{MC} \mid B)P(B)}{P(\text{MC} \mid A)P(A) + P(\text{MC} \mid B)P(B) + P(\text{MC} \mid C)P(C)}$$

$$= \frac{1 \times \frac{1}{3}}{\frac{1}{2} \times \frac{1}{3} + 1 \times \frac{1}{3} + 0 \times \frac{1}{3}} = \frac{2}{3}.$$

Thus, switching doors will give a probability of winning a car that is twice as great as staying with the initial choice.

6.2.4 PREDICTING RACE USING SURNAME AND RESIDENCE LOCATION

This section contains an advanced application of conditional probability and Bayes' rule in the social sciences. Readers may skip this section without affecting their ability to understand the materials in the remainder of the book.

It is often of interest to infer certain unknown attributes of individuals from their known characteristics. We consider the problem of predicting individual race using surname and residence location.[2] Accurate prediction of individual race is useful, for example, when studying turnout rates among racial groups.

The US Census Bureau releases a list of common surnames with their frequency. For example, the most common surname was "Smith" with 2,376,206 occurrences, followed by "Johnson" and "Williams" with 1,857,160 and 1,534,042, respectively. This data set is quite comprehensive, including a total of more than 150,000 surnames that occurred at least 100 times. In addition, the census provides the relative frequencies of individual race within each surname, using a six-category self-reported race measure: non-Hispanic white, non-Hispanic black, non-Hispanic Asian and Pacific Islander, Hispanic origin, non-Hispanic American Indian and Alaskan Native, and non-Hispanic of two or more races. We will combine the last two categories into a single category of non-Hispanic others, so that we have five categories in total. The aggregate information, which can be written as $P(\textsf{race} \mid \textsf{surname})$, enables us to predict race given an individual's surname.

Note that $P(\textsf{race})$, $P(\textsf{race} \mid \textsf{surname})$, and $P(\textsf{race and surname})$ are examples of general ways to represent the marginal, conditional, and joint probabilities, respectively. For example, $P(\textsf{race})$ represents a collection of marginal probabilities, i.e., $P(\textsf{white})$, $P(\textsf{black})$, $P(\textsf{asian})$, $P(\textsf{hispanic})$, and $P(\textsf{others})$. Similarly, $P(\textsf{race} \mid \textsf{surname})$ can be evaluated for any given racial group and surname, for example, $P(\textsf{black} \mid \textsf{Smith})$. To illustrate the convenience of this general notation, we apply the law of total probability in equation (6.14) to the joint probability of race and surname:

$$P(\textsf{surname}) = \sum_{\textsf{race}} P(\textsf{race and surname}),$$

where the summation is taken over all racial categories (i.e., white, black, asian, hispanic, and others. In terms of the notation used in equation (6.14), A represents any given surname while B_i is a racial category. This equality applies to any surname of interest, and the summation is taken over all five racial categories.

This census name list is contained in the CSV data file `names.csv`. Table 6.3 lists the names and descriptions of variables in this census surname list data set.[3]

[2] This section is in part based on Kosuke Imai and Kabir Khanna (2016) "Improving ecological inference by predicting individual ethnicity from voter registration records." *Political Analysis*, vol. 24, no. 2 (Spring), pp. 263–272.

[3] To protect anonymity, the Census Bureau does not reveal small race percentages for given surnames. For the sake of simplicity, we impute these missing values by assuming that residual values will be equally allocated to the racial categories with missing values. That is, for each last name, we subtract the sum of the percentages of all races without missing values from 100% and divide the remaining percentage equally among those races that do have missing values.

Table 6.3. US Census Bureau Surname List Data.

Variable	*Description*
`surname`	surname
`count`	number of individuals with a specific surname
`pctwhite`	percentage of non-Hispanic whites among those who have a specific surname
`pctblack`	percentage of non-Hispanic blacks among those who have a specific surname
`pctapi`	percentage of non-Hispanic Asians and Pacific Islanders among those who have a specific surname
`pcthispanic`	percentage of Hispanic origin among those who have a specific surname
`pctothers`	percentage of the other racial groups among those who have a specific surname

```
cnames <- read.csv("names.csv")
dim(cnames)
## [1] 151671      7
```

The total number of surnames contained in this data set is 151,671. For these surnames, the data set gives the probability of belonging to a particular racial group given a voter's surname, i.e., $P(\text{race} \mid \text{surname})$. We begin by using this conditional probability to classify the race of individual voters. To validate the accuracy of our prediction of individual race, we use the sample of 10,000 registered voters from Florida analyzed earlier (see table 6.1). In some Southern states including Florida, voters are asked to self-report their race when registering. This makes the Florida data an ideal validation data set. If the accuracy of a prediction method is empirically validated in Florida, we may use the method to predict individual race in other states where such information is not available.

For matching names between the voter file and census name data, we use the `match()` function. This function takes the syntax of `match(x, y)` and returns a vector of indices of vector `y`'s correspondence to each element of vector `x`. The function returns `NA` if there is no match found in `y` for an element of `x`. Here is a simple example illustrating the use of the `match()` function.

```
x <- c("blue", "red", "yellow")
y <- c("orange", "blue")
## match x with y
match(x, y) # "blue" appears in the 2nd element of y
## [1]  2 NA NA
## match y with x
```

```
match(y, x) # "blue" appears in the first element of x

## [1] NA  1
```

Going back to the problem of predicting individual racial groups, we remove voters whose surnames do not appear in the census surname list. To do so, we utilize the fact that the syntax `match(x, y)` returns `NA` if the corresponding element of `x` is not matched with any element of `y`.

```
FLVoters <- FLVoters[!is.na(match(FLVoters$surname, cnames$surname)), ]
dim(FLVoters)

## [1] 8022    7
```

The syntax `!is.na()` represents "not `NA`," where `!` indicates negation, so that only the matched elements are retained. Thus, we focus on the resulting 80% of the original sample. We first compute the proportion of voters whose race is correctly classified in each racial category. Race is considered correctly classified if the racial category with the greatest conditional probability $P(\text{race} \mid \text{surname})$ is identical to the self-reported race. These represent *true positives* of classification (see table 4.3).

We calculate the *true positive rate* for each racial group, which represents, for example, the proportion of white voters who are correctly predicted as white. To compute this, we first subset white voters from the Florida voter file and then match the surname of each voter with the same surname in the census surname data.

```
whites <- subset(FLVoters, subset = (race == "white"))
w.indx <- match(whites$surname, cnames$surname)
head(w.indx)

## [1]  8610   237  4131  2244 27852  3495
```

The outputted row index `w.indx` contains, for each observation in the `whites` data frame, the number of the row with the same surname in the `cnames` data frame. For example, the second observation in the `whites` data frame has the surname Lynch. This surname appears in the 237th row of the `cnames` data set. Accordingly, the second value in `w.indx` is 237. More specifically, for each surname belonging to a white voter in Florida, we use `apply(cnames[w.indx, vars], 1, max)` to compare the predicted probabilities across the five racial categories in the vector `vars`, and extract the highest predicted probability. We then check whether the highest predicted probability for that voter is the same as the predicted probability of their being white. If these two numbers are identical, the classification is correct. Finally, we compute the mean of the resulting binary vector to obtain the proportion of correct classifications, the true positive rate.

```r
## relevant variables
vars <- c("pctwhite", "pctblack", "pctapi", "pcthispanic", "pctothers")
mean(apply(cnames[w.indx, vars], 1, max) == cnames$pctwhite[w.indx])

## [1] 0.950218
```

The result shows that 95% of white voters are correctly predicted as whites. We repeat the same analysis for black, Hispanic, and Asian voters.

```r
## black
blacks <- subset(FLVoters, subset = (race == "black"))
b.indx <- match(blacks$surname, cnames$surname)
mean(apply(cnames[b.indx, vars], 1, max) == cnames$pctblack[b.indx])

## [1] 0.1604824

## Hispanic
hispanics <- subset(FLVoters, subset = (race == "hispanic"))
h.indx <- match(hispanics$surname, cnames$surname)
mean(apply(cnames[h.indx, vars], 1, max) == cnames$pcthispanic[h.indx])

## [1] 0.8465298

## Asian
asians <- subset(FLVoters, subset = (race == "asian"))
a.indx <- match(asians$surname, cnames$surname)
mean(apply(cnames[a.indx, vars], 1, max) == cnames$pctapi[a.indx])

## [1] 0.5642857
```

We find that surname alone can correctly classify 85% of Hispanic voters as Hispanic. In contrast, classification of Asian and black voters is much worse. In particular, only 16% of black voters are correctly classified as African-Americans. The high true positive rate for whites may simply arise from the fact that they far outnumber voters from other racial categories.

We next look at *false positives*. Below, we calculate the *false discovery rate* for each racial group, which, for example, represents the proportion of voters who are not white among those classified as white. We use the same indexing trick as above and compute the proportion of white voters among those classified as whites. Subtracting the resulting value from 1 yields the false discovery rate for whites.

```r
indx <- match(FLVoters$surname, cnames$surname)
## white false discovery rate
1 - mean(FLVoters$race[apply(cnames[indx, vars], 1, max) ==
                         cnames$pctwhite[indx]] == "white")

## [1] 0.1973603
```

Table 6.4. Florida Census Data at the Voting District Level.

Variable	*Description*
`county`	county census ID of the voting district
`VTD`	voting district census ID (only unique within the county)
`total.pop`	total population of the voting district
`white`	proportion of non-Hispanic whites in the voting district
`black`	proportion of non-Hispanic blacks in the voting district
`api`	proportion of non-Hispanic Asians and Pacific Islanders in the voting district
`hispanic`	proportion of voters of Hispanic origin in the voting district
`others`	proportion of the other racial groups in the voting district

```
## black false discovery rate
1 - mean(FLVoters$race[apply(cnames[indx, vars], 1, max) ==
                          cnames$pctblack[indx]] == "black")

## [1] 0.3294574

## Hispanic false discovery rate
1 - mean(FLVoters$race[apply(cnames[indx, vars], 1, max) ==
                          cnames$pcthispanic[indx]] == "hispanic")

## [1] 0.2274755

## Asian false discovery rate
1 - mean(FLVoters$race[apply(cnames[indx, vars], 1, max) ==
                          cnames$pctapi[indx]] == "asian")

## [1] 0.3416667
```

The results show that the false discovery rate is the highest for Asian and black voters, while it is much lower for whites and Hispanics.

Next, we attempt to improve the above prediction by taking into account where voters live. This approach should be helpful to the extent that there exists residential segregation based on race. In the United States, voter files contain voters' addresses. Using this information, our data set also provides the voting district where each voter lives. In addition, we will utilize the Florida census data, which contains the racial composition of each voting district. The names and descriptions of variables in this census data set, `FLCensusVTD.csv`, are given in table 6.4.

How does the knowledge of residence location improve the prediction of individual race? Whereas the census name data set contains information about the conditional probability $P(\text{race} \mid \text{surname})$, the Florida census data set provides additional information about $P(\text{race} \mid \text{residence})$ (proportion of each racial category among residents in a given voting district) and $P(\text{race})$ (proportion of residents who live in a given voting district). We wish to combine them and compute the desired conditional probability $P(\text{race} \mid \text{surname and residence})$. Recall that these are general ways to represent marginal, conditional, and joint probabilities. Each expression can be evaluated using a specific racial group, surname, and residential location.

Computing $P(\text{race} \mid \text{surname and residence})$ requires Bayes' rule. So far, we have employed Bayes' rule for one event A conditional on an event B, but now we need to use Bayes' rule conditional on both B and another event C:

$$P(A \mid B, C) = \frac{P(B \mid A \text{ and } C)P(A \mid C)}{P(B \mid C)},$$

where every probability on the right-hand side is defined conditional on another event C (see equation (6.21)). Applying this rule yields

$$P(\text{race} \mid \text{surname and residence}) \\ = \frac{P(\text{surname} \mid \text{race and residence})P(\text{race} \mid \text{residence})}{P(\text{surname} \mid \text{residence})}. \quad (6.22)$$

In this equation, while $P(\text{race} \mid \text{residence})$ is available from the Florida census data, the other two conditional probabilities, $P(\text{surname} \mid \text{race and residence})$ and $P(\text{surname} \mid \text{residence})$, are not directly given either in the census name data set or the Florida census data set.

To overcome this difficulty, we make an additional assumption that a voter's surname and residence location are independent of each other, given race. This *conditional independence* assumption implies that once we know a voter's race, their residence location does not give us any additional information about their surname. So long as there is no strong geographical concentration of certain surnames in Florida within a racial category, this assumption is reasonable. The assumption is violated, for example, if Hispanic Cubans tend to have distinct names and are concentrated in certain neighborhoods. Unfortunately, our data cannot tell us whether this assumption is appropriate, but we will proceed assuming it is. Applying equation (6.20), the assumption can be written as

$$\begin{aligned} P(\text{surname} \mid \text{race and residence}) &= \frac{P(\text{surname and race} \mid \text{residence})}{P(\text{race} \mid \text{residence})} \\ &= \frac{P(\text{surname} \mid \text{residence})P(\text{race} \mid \text{residence})}{P(\text{race} \mid \text{residence})} \\ &= P(\text{surname} \mid \text{race}). \qquad (6.23) \end{aligned}$$

The first equality follows from the definition of conditional probability, whereas the second equality is due to the application of equation (6.20).

The assumption transforms equation (6.22) into

$$P(\text{race} \mid \text{surname and residence}) = \frac{P(\text{surname} \mid \text{race})P(\text{race} \mid \text{residence})}{P(\text{surname} \mid \text{residence})}.$$

We should keep this key version of the equation in mind as the one we will ultimately use.

Note that applying the law of total probability defined in equation (6.14) and then invoking the assumption given in equation (6.23), the denominator of equation (6.22) can be written as the following equation, which sums over all racial categories:

$$\begin{aligned} P(\textsf{surname} \mid \textsf{residence}) &= \sum_{\textsf{race}} P(\textsf{surname} \mid \textsf{race and residence}) P(\textsf{race} \mid \textsf{residence}) \\ &= \sum_{\textsf{race}} P(\textsf{surname} \mid \textsf{race}) P(\textsf{race} \mid \textsf{residence}). \end{aligned} \tag{6.24}$$

In the above equations, we use $\sum_{\textsf{race}}$ to indicate summation over all categories of the race variable (i.e., black, white, Asian, Hispanic, and others).

While the census surname list gives $P(\textsf{race} \mid \textsf{surname})$, the prediction of individual race based on equation (6.22) requires the computation of $P(\textsf{surname} \mid \textsf{race})$, which is included in both the numerator and the denominator (see equation (6.24)). Fortunately, we can use Bayes' rule to obtain

$$P(\textsf{surname} \mid \textsf{race}) = \frac{P(\textsf{race} \mid \textsf{surname}) P(\textsf{surname})}{P(\textsf{race})}. \tag{6.25}$$

The two terms in the numerator of equation (6.25) can be computed using the census name list. We compute $P(\textsf{race})$, which is not included in that data, from the Florida census data by using the law of total probability:

$$P(\textsf{race}) = \sum_{\textsf{residence}} P(\textsf{race} \mid \textsf{residence}) P(\textsf{residence}). \tag{6.26}$$

In this equation, $\sum_{\textsf{residence}}$ indicates summation over all values of the residence variable (i.e., all voting districts in the data).

To implement this prediction methodology in R, we first compute $P(\textsf{race})$ using equation (6.26). We do so by calculating a *weighted average* of percentages for each racial category across voting districts with the population of the voting district, which is proportional to $P(\textsf{residence})$, as the weight. The `weighted.mean()` function can be used to compute weighted averages, in which the `weights` argument takes a vector of weights.

```
FLCensus <- read.csv("FLCensusVTD.csv")
## compute proportions by applying weighted.mean() to each column
race.prop <-
    apply(FLCensus[, c("white", "black", "api", "hispanic", "others")],
          2, weighted.mean, weights = FLCensus$total.pop)
race.prop # race proportions in Florida

##      white      black        api   hispanic     others
## 0.60451586 0.13941679 0.02186662 0.21279972 0.02140101
```

We can now compute $P(\text{surname} \mid \text{race})$ using equation (6.25) and the census name list.

```
total.count <- sum(cnames$count)
## P(surname | race) = P(race | surname) * P(surname) / P(race)
cnames$name.white <- (cnames$pctwhite / 100) *
    (cnames$count / total.count) / race.prop["white"]
cnames$name.black <- (cnames$pctblack / 100) *
    (cnames$count / total.count) / race.prop["black"]
cnames$name.hispanic <- (cnames$pcthispanic / 100) *
    (cnames$count / total.count) / race.prop["hispanic"]
cnames$name.asian <- (cnames$pctapi / 100) *
    (cnames$count / total.count) / race.prop["api"]
cnames$name.others <- (cnames$pctothers / 100) *
    (cnames$count / total.count) / race.prop["others"]
```

Next, we compute the denominator of equation (6.22), $P(\text{surname} \mid \text{residence})$, using equation (6.24). To do this, we merge the census data into the voter file data using the `county` and `VTD` variables. In the `merge()` function, we set the `all` argument to `FALSE` so that nonmatching rows in both data sets will be dropped (see section 4.2.5). Since the census data includes $P(\text{race} \mid \text{residence})$ as a variable for each racial category, the merged data set will as well.

```
FLVoters <- merge(x = FLVoters, y = FLCensus, by = c("county", "VTD"),
                  all = FALSE)
## P(surname | residence) = sum_race P(surname | race) P(race | residence)
indx <- match(FLVoters$surname, cnames$surname)
FLVoters$name.residence <- cnames$name.white[indx] * FLVoters$white +
    cnames$name.black[indx] * FLVoters$black +
    cnames$name.hispanic[indx] * FLVoters$hispanic +
    cnames$name.asian[indx] * FLVoters$api +
    cnames$name.others[indx] * FLVoters$others
```

We have now calculated every quantity contained in our key version of equation (6.22): $P(\text{surname} \mid \text{race})$, $P(\text{race} \mid \text{residence})$, and $P(\text{surname} \mid \text{residence})$. Finally, we plug the quantities into the equation to compute the predicted probability that an individual belongs to a particular race, given his or her surname and residence.

```
## P(race | surname, residence) = P(surname | race) * P(race | residence)
##                                 / P(surname | residence)
FLVoters$pre.white <- cnames$name.white[indx] * FLVoters$white /
    FLVoters$name.residence
```

```
FLVoters$pre.black <- cnames$name.black[indx] * FLVoters$black /
    FLVoters$name.residence
FLVoters$pre.hispanic <- cnames$name.hispanic[indx] * FLVoters$hispanic /
    FLVoters$name.residence
FLVoters$pre.asian <- cnames$name.asian[indx] * FLVoters$api /
    FLVoters$name.residence
FLVoters$pre.others <- 1 - FLVoters$pre.white - FLVoters$pre.black -
    FLVoters$pre.hispanic - FLVoters$pre.asian
```

We evaluate the accuracy of this prediction methodology and assess how much improvement knowledge of the voters' location of residence yields. We begin by examining true positives for each race using the same programming trick as before.

```
## relevant variables
vars1 <- c("pre.white", "pre.black", "pre.hispanic", "pre.asian",
           "pre.others")
## white
whites <- subset(FLVoters, subset = (race == "white"))
mean(apply(whites[, vars1], 1, max) == whites$pre.white)

## [1] 0.9371366

## black
blacks <- subset(FLVoters, subset = (race == "black"))
mean(apply(blacks[, vars1], 1, max) == blacks$pre.black)

## [1] 0.6474954

## Hispanic
hispanics <- subset(FLVoters, subset = (race == "hispanic"))
mean(apply(hispanics[, vars1], 1, max) == hispanics$pre.hispanic)

## [1] 0.85826

## Asian
asians <- subset(FLVoters, subset = (race == "asian"))
mean(apply(asians[, vars1], 1, max) == asians$pre.asian)

## [1] 0.6071429
```

The true positive rate for blacks has jumped from 16% to 65%. Minor improvements are also made for Hispanic and Asian voters. Since African-Americans tend to live close to one another in the United States, the location of voters' residences can be informative. For example, according to the census data, among people whose surname is "White," 27% are black. However, once we incorporate the location of their residence, the predicted probability of such individuals being black ranges from 1% to 98%. This implies that we predict some voters to be highly likely black and others highly likely nonblack.

```
## proportion of blacks among those with surname "White"
cnames$pctblack[cnames$surname == "WHITE"]

## [1] 27.38

## predicted probability of being black given residence location
summary(FLVoters$pre.black[FLVoters$surname == "WHITE"])

##      Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.005207 0.081150 0.176300 0.264000 0.320000 0.983700
```

Finally, we compute the false positive rate for each race.

```
## white
1 - mean(FLVoters$race[apply(FLVoters[, vars1], 1, max)==
                             FLVoters$pre.white] == "white")

## [1] 0.1187425

## black
1 - mean(FLVoters$race[apply(FLVoters[, vars1], 1, max)==
                             FLVoters$pre.black] == "black")

## [1] 0.2346491

## Hispanic
1 - mean(FLVoters$race[apply(FLVoters[, vars1], 1, max) ==
                             FLVoters$pre.hispanic] == "hispanic")

## [1] 0.2153709

## Asian
1 - mean(FLVoters$race[apply(FLVoters[, vars1], 1, max) ==
                             FLVoters$pre.asian] == "asian")

## [1] 0.3461538
```

We find that the false positive rate for whites is significantly reduced. This is in large part due to the fact that many of the black voters who were incorrectly classified as whites using surname alone are now predicted to be black. In addition, the false positive rate for blacks lowered by a similar amount. This example illustrates the powerful use of conditional probability and Bayes' rule.

6.3 Random Variables and Probability Distributions

We have so far considered various events including a coin landing on heads, twins being both boys, and a voter being African-American. In this section, we introduce the concept of *random variables* and their *probability distributions*, which further widens the scope of mathematical analyses of these events.

6.3.1 RANDOM VARIABLES

A random variable assigns a number to each event. For example, two outcomes of a coin flip can be represented by a binary random variable where 1 indicates landing on heads and 0 denotes landing on tails. Another example is one's income measured in dollars. The values of random variables must represent *mutually exclusive and exhaustive* events. That is, different values cannot represent the same event and all events should be represented by some values. For example, consider a random variable that represents one's racial group using five unique integers: black = 1, white = 2, hispanic = 3, asian = 4, and others = 5. According to this definition, someone who self-identifies as black and white will be assigned the value of 5 instead of taking the values of 1 and 2 at the same time.

There are two types of random variables, depending on the type of values they take. The first is a *discrete random variable*, which takes a finite (or at most countably infinite) number of distinct values. Examples include categorical or factor variables such as racial groups and number of years of education. The second type is a *continuous random variable*, which takes a value within an interval of the real line. That is, the variable can assume uncountably many values. Examples of continuous random variables include height, weight, and gross domestic product (GDP). The use of random variables, instead of events, facilitates the development of mathematical rules for probability because a random variable takes numeric values. Once we define a random variable, we can formalize a *probability model* using the distribution of the random variable.

A **random variable** assigns a numeric value to each event of the experiment. These values represent mutually exclusive and exhaustive events, together forming the entire sample space. A **discrete random variable** takes a finite or at most countably infinite number of distinct values, whereas a **continuous random variable** assumes an uncountably infinite number of values.

6.3.2 BERNOULLI AND UNIFORM DISTRIBUTIONS

We first consider the simplest example of a *discrete random variable*: a coin flip. For this experiment, we define a *binary random variable* X, which is equal to 1 if a coin lands on heads, and 0 otherwise. In general, a random variable that takes two distinct values is called a *Bernoulli random variable*. Notice that this setup applies to any experiment with two distinct events. Examples include {vote, abstain}, {win election, lose election}, and {correct classification, misclassification}. Thus, whether a voter turns out ($X = 1$) or not ($X = 0$) can be represented by a Bernoulli random variable. Generically, we consider the event $X = 1$ a success and the event $X = 0$ a failure. We use p to denote the probability of success.

The distribution of a discrete random variable can be characterized by the *probability mass function* (*PMF*). The PMF $f(x)$ of a random variable X is defined as the probability that the random variable takes a particular value x, i.e., $f(x) = P(X = x)$. That is, given the input x, which is a specific value of choice, the PMF $f(x)$ returns as

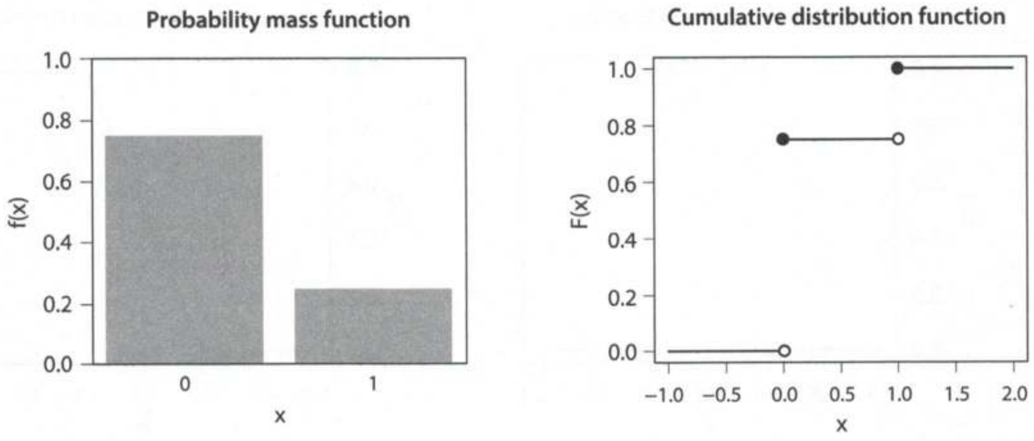


Figure 6.5. The Probability Mass and Cumulative Distribution Functions for a Bernoulli Random Variable. The probability of success is 0.25. The open and solid circles represent the exclusion and inclusion of the corresponding points, respectively.

the output the probability that a random variable X takes that value x. In the case of a Bernoulli random variable, the PMF takes the value of p when $x = 1$ and that of $1 - p$ when $x = 0$. The function is zero at all other values of x.

Another important function related to probability distribution is the *cumulative distribution function* (*CDF*). The CDF $F(x)$ represents the cumulative probability that a random variable X takes a value equal to or less than a specific value x, i.e., $F(x) = P(X \leq x)$. The CDF, therefore, represents the sum of the PMF $f(x)$ evaluated at all values up to x. Formally, the relationship between the PMF $f(x)$ and the CDF $F(x)$ for a discrete random variable can be written as

$$F(x) = P(X \leq x) = \sum_{k \leq x} f(k),$$

where k represents all values the random variable X can take that are less than or equal to x. That is, the CDF equals the sum of the PMFs. The CDF ranges from 0 to 1 for any random variable, whether continuous or discrete. It is a nondecreasing function because as x increases, more probability will be added.

The CDF $F(x)$ for a Bernoulli random variable is simple. It is zero for all negative values of x because the random variable never assumes any of those values. The CDF then takes the value of $1 - p$ when $x = 0$, which is the probability that X equals 0. The function stays flat at $1 - p$ when $0 \leq x < 1$ because none of these values will be realized. At $x = 1$, the CDF equals 1 because the random variable takes either the value of 0 or 1, and stays at this value when $x \geq 1$ because X does not take any value greater than 1. Figure 6.5 graphically displays the PMF and CDF of a Bernoulli random variable when $p = 0.25$. The open and solid circles represent the exclusion and inclusion of the corresponding points, respectively.

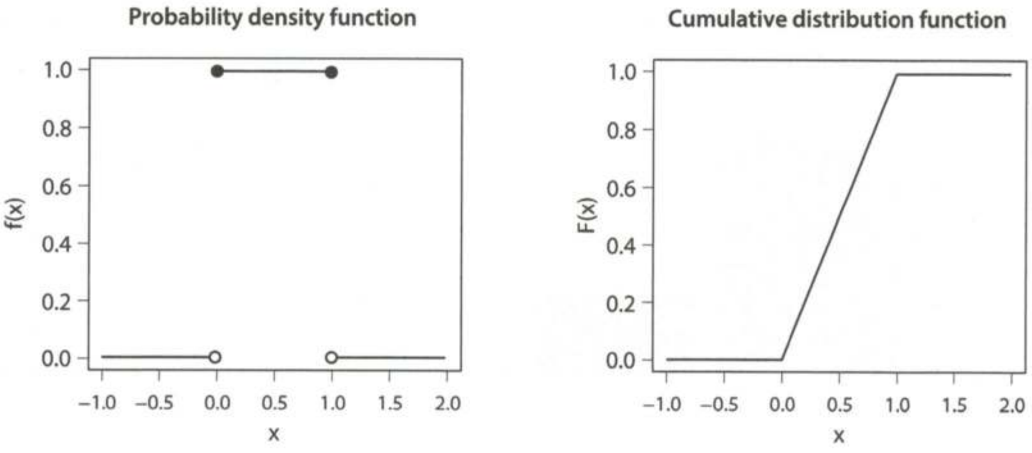


Figure 6.6. The Probability Density and Cumulative Distribution Functions for a Uniform Random Variable. The interval is set to $[0, 1]$. The open and solid circles represent the exclusion and inclusion of the corresponding points, respectively.

The **probability mass function** (PMF) of a **Bernoulli random variable** with success probability p is given by

$$f(x) = \begin{cases} p & \text{if } x = 1, \\ 1 - p & \text{if } x = 0, \\ 0 & \text{otherwise,} \end{cases}$$

where $f(1)$ and $f(0)$ represent the probability of success and failure, respectively. The **cumulative distribution function** (CDF) is given by

$$F(x) = \begin{cases} 0 & \text{if } x < 0, \\ 1 - p & \text{if } 0 \leq x < 1, \\ 1 & \text{if } x \geq 1. \end{cases}$$

We now discuss a *uniform random variable* as a simple example of a *continuous random variable*. A uniform random variable takes every value within a given interval $[a, b]$ with equal likelihood. The PMF is not defined for a continuous random variable because this variable assumes an uncountably infinite number of values. Instead, we use the *probability density function (PDF)* $f(x)$ (or simply, density function), which quantifies the likelihood that a continuous random variable X will take a specific value x. We have already seen the concept of *density*, which is used to measure the height of bins in a histogram (see section 3.3.2). The value of the PDF is nonnegative and can be greater than 1. Moreover, like density in histograms, the area under the PDF must sum to 1.

Since each value within the interval is equally likely to be realized, the PDF for the uniform distribution is a flat horizontal line defined by $1/(b - a)$. In other words, the PDF does not depend on x and always equals $1/(b - a)$ within the interval. The height is determined so that the area below the line equals 1 as required. The left-hand plot of figure 6.6 graphically displays the PDF for a uniform distribution when the interval is set to $[0, 1]$.

We can also define the *cumulative distribution function* (CDF) for a continuous random variable. The definition of the CDF is the same as the case of discrete random variables. That is, the CDF $F(x)$ represents the probability that a random variable X takes a value less than or equal to a specific value x, i.e., $P(X \leq x)$. Graphically, the CDF corresponds to the area under the probability density function curve up to the value x (from negative infinity). Mathematically, this notion can be expressed using integration instead of summation:

$$F(x) = P(X \leq x) = \int_{-\infty}^{x} f(t)\, dt.$$

Since the entire area under the probability density curve has to sum to 1, we have $F(x) = 1$ when $x = \infty$. The CDF for the uniform distribution is shown in the right-hand plot of figure 6.6. In this case, the CDF is a straight line, as shown in the right-hand plot of the figure, because the area under the PDF increases at a constant rate.

The **probability density function** (PDF) of a **uniform random variable** with interval $[a, b]$ is given by

$$f(x) = \begin{cases} \frac{1}{b-a} & \text{if } a \leq x \leq b, \\ 0 & \text{otherwise.} \end{cases}$$

The **cumulative probability function** (CDF) is given by

$$F(x) = \begin{cases} 0 & \text{if } x < a, \\ \frac{x-a}{b-a} & \text{if } a \leq x < b, \\ 1 & \text{if } x \geq b. \end{cases}$$

We can easily compute the PDF and CDF of a uniform distribution in R. For the PDF $f(x)$, we use the `dunif()` function where the main argument is the value x at which the function is evaluated and the interval is specified using the `min` and `max` arguments. We can compute the CDF in a similar manner using the `punif()` function. The `d` in `dunif()` indicates density, whereas the `p` in `punif()` stands for probability.

```
## uniform PDF: x = 0.5, interval = [0, 1]
dunif(0.5, min = 0, max = 1)

## [1] 1

## uniform CDF: x = 1, interval = [-2, 2]
punif(1, min = -2, max = 2)

## [1] 0.75
```

The two distributions we have introduced here share a useful connection. We can use a uniform random variable to generate a Bernoulli random variable. To do this, notice that under the uniform distribution with unit interval [0, 1], the CDF is given by the 45-degree line, i.e., $F(x) = x$. Therefore, the probability that this uniform random variable X takes a value less than or equal to x is equal to x when $0 \leq x \leq 1$. Thus, in order to generate a Bernoulli random variable Y with success probability p, we can first sample a uniform random variable X and then set $Y = 1$ when X is less than p (similarly, set $Y = 0$ if $X \geq p$) so that Y takes a value of 1 with probability p. To do this *Monte Carlo simulation* in R, we use the `runif()` function to generate a uniform random variable by setting the `min` and `max` arguments to `0` and `1`, respectively.

```
sims <- 1000
p <- 0.5 # success probabilities
x <- runif(sims, min = 0, max = 1) # uniform [0, 1]
head(x)

## [1] 0.292614295 0.619951024 0.004618747 0.162426728
## [5] 0.001157040 0.655518809

y <- as.integer(x <= p) # Bernoulli; turn TRUE/FALSE to 1/0
head(y)

## [1] 1 0 1 1 1 0

mean(y) # close to success probability p, proportion of 1s vs. 0s

## [1] 0.521
```

6.3.3 BINOMIAL DISTRIBUTION

The *binomial distribution* is a generalization of the Bernoulli distribution. Instead of a single coin flip, we consider an experiment in which the same coin is flipped independently and multiple times. That is, a binomial random variable can represent the number of times a coin lands on heads in multiple trials of independent coin flips.

More generally, a binomial random variable X records the number of successes in a total of n independent and identical trials with success probability p. In other words, a binomial random variable is the sum of n *independently and identically distributed* (or *i.i.d.* in short) Bernoulli random variables. Recall that a Bernoulli random variable equals either 1 or 0 with success probability p. Thus, X can take an integer value from 0 to n. Since the binomial distribution is discrete, its PMF can be interpreted as the probability of X taking a specific value x. The CDF represents the cumulative probability that a binomial random variable has x or fewer successes out of n trials. The PMF and CDF of a binomial random variable are given by the following formulas, which involve combinations (see equation (6.10)). No simple expression exists for the CDF, which is written as the sum of the PMFs.

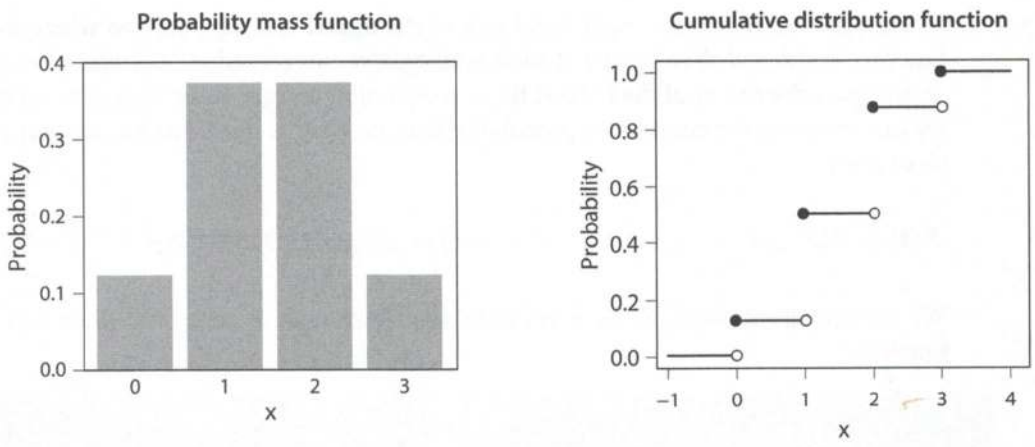


Figure 6.7. The Probability Mass and Cumulative Distribution Functions for a Binomial Random Variable. The success probability is 0.5 and the total number of trials is 3. The open and solid circles represent the exclusion and inclusion of the corresponding points, respectively. Source: Adapted from example by Paul Gaborit, http://texample.net.

The probability mass function (PMF) of a **binomial random variable** with success probability p and n trials is given by

$$f(x) = P(X = x) = \binom{n}{x} p^x (1-p)^{n-x}. \tag{6.27}$$

The cumulative distribution function (CDF) can be written as

$$F(x) = P(X \leq x) = \sum_{k=0}^{x} \binom{n}{k} p^k (1-p)^{n-k},$$

for $x = 0, 1, \ldots, n$.

Figure 6.7 shows the PMF and CDF when $p = 0.5$ and $n = 3$. For example, we can compute the probability that we obtain two successes out of three trials, which is the height of the third bar in the left-hand plot of the figure:

$$f(2) = P(X = 2) = \binom{3}{2} \times 0.5^2 \times (1-0.5)^{3-2} = \frac{3!}{(3-2)!2!} \times 0.5^3 = 0.375.$$

Calculating the PMF of a binomial distribution is straightforward. The `dbinom()` function takes the number of successes as the main argument, and the `size` and `prob` arguments specify the number of trials and success probability, respectively.

```
## PMF when x = 2, n = 3, p = 0.5
dbinom(2, size = 3, prob = 0.5)

## [1] 0.375
```

The CDF, shown in the right-hand plot of the figure, is a *step function* where the function is flat and then jumps at each nonnegative integer value. The size of each jump equals the height of the PMF at the corresponding integer value. Using the CDF, we can compute the cumulative probability that we have at most one success out of three trials:

$$F(1) = P(X \leq 1) = P(X = 0) + P(X = 1) = f(0) + f(1) = 0.125 + 0.375 = 0.5.$$

We can compute the CDF of a binomial distribution in R using the pbinom() function.

```
## CDF when x = 1, n = 3, p = 0.5
pbinom(1, size = 3, prob = 0.5)

## [1] 0.5
```

An intuitive explanation covers why the PMF of a binomial distribution looks like equation (6.27). When we flip a coin n times, each unique sequence of n outcomes is equally likely. For example, if $n = 5$, then the event that only the last two coin flips land on tails $\{HHHTT\}$ is equally as likely as the event that the flips alternate landing on heads and tails $\{HTHTH\}$, where we use H and T to denote the events that a coin lands on heads and tails, respectively. However, for the binomial distribution only the number of heads matters. As a result, these two events represent the same outcome. We use combinations to count the number of ways we can have x successes out of n trials, which is equal to ${}_nC_x = \binom{n}{x}$. We multiply this by the probability of x successes, which is equal to p^x (because each trial is independent), and the probability of $n - x$ failures, which is given by $(1 - p)^{n-x}$ (again because of independence).

As an application of the binomial distribution, consider the probability that one's vote is pivotal in an election. Your vote is pivotal if the election is tied before you cast your ballot. Suppose that in a large population exactly 50% of voters support an incumbent while the other half support a challenger. Further, assume that whether voters turn out or not has nothing to do with their vote choice. Under this scenario, what is the probability that the election ends up with an exact tie? We compute this probability when the number of voters who turn out equals 1000, then 10,000, and then 100,000. To compute this probability, we can evaluate the PMF of the binomial distribution by setting the success probability to 50% and the size to the total number of voters who turn out. We then evaluate the PMF at exactly half of all voters who turn out. We find that the probability of a tie is quite small, even when the population of voters is evenly divided.

```
## number of voters who turn out
voters <- c(1000, 10000, 100000)
dbinom(voters / 2, size = voters, prob = 0.5)

## [1] 0.025225018 0.007978646 0.002523126
```

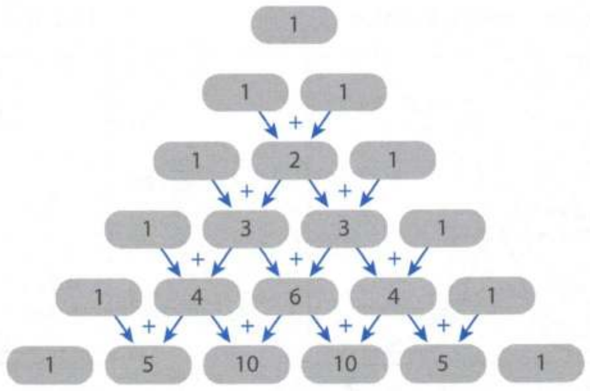


Figure 6.8. Pascal's Triangle. Binomial coefficients can be represented as Pascal's triangle, where the xth element of the nth row returns the binomial coefficient $\binom{n-1}{x-1}$. Source: Adapted from example by Paul Gaborit, http://texample.net.

Where does the name "binomial distribution" come from? The name of this distribution is based on the following *binomial theorem*.

> The **binomial theorem** shows how to compute the coefficient of each term when expanding the power of a binomial, i.e., $(a + b)^n$. That is, the coefficient for the term $a^x b^{n-x}$ when expanding $(a + b)^n$ is equal to $\binom{n}{x}$.

For example, according to the binomial theorem, when $n = 4$, the coefficient for the term a^2b^2 when expanding $(a + b)^4$ is equal to $\binom{4}{2} = 6$. This result is confirmed by writing out the entire expansion:

$$(a + b)^4 = a^4 + 4a^3b + 6a^2b^2 + 4ab^3 + b^4. \tag{6.28}$$

These binomial coefficients can be organized as *Pascal's triangle*, as shown in figure 6.8. For example, the coefficients for the terms resulting from the expansion of $(a + b)^4$ in equation (6.28) are shown in the fifth row of Pascal's triangle. More generally, in Pascal's triangle, the xth element of the nth row represents the binomial coefficient $\binom{n-1}{x-1}$. In addition, as shown in the figure, each element equals the sum of the two elements just above it, leading to a straightforward sequential computation of binomial coefficients. This makes sense because, for example, $(a + b)^4$ can be written as the product of $(a + b)^3$ and $(a + b)$,

$$(a + b)^4 = (a^3 + 3a^2b + 3ab^2 + b^3)(a + b).$$

In this example, the coefficient for a^2b^2 is based on the sum of two products, i.e., $3a^2b \times b$ and $3ab^2 \times a$, and hence is equal to $6 = 3 + 3$. In general, to obtain x

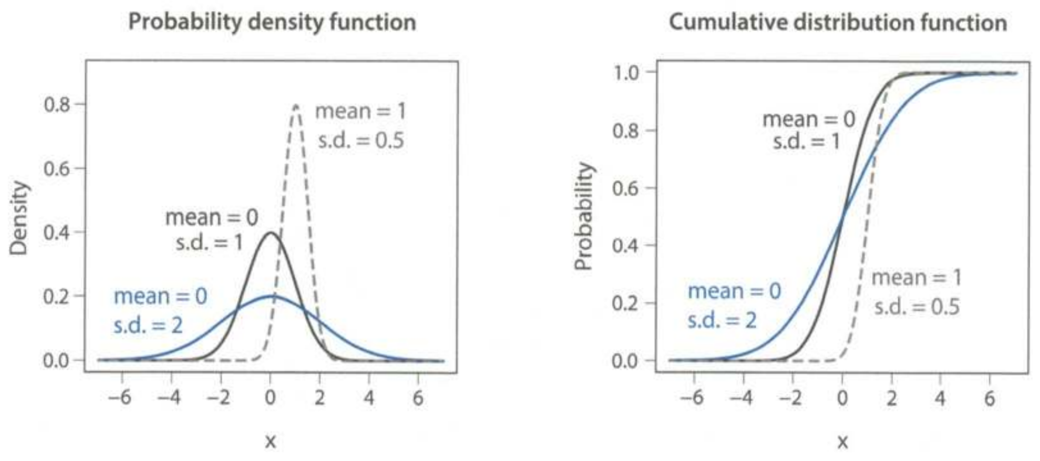


Figure 6.9. The Probability Density and Cumulative Distribution Functions of the Normal Distribution.

success combinations out of n trials, we consider two scenarios—the last trial ending in a success or ending in a failure—and add the total number of combinations under each scenario:

$$\begin{aligned}\binom{n-1}{x} + \binom{n-1}{x-1} &= \frac{(n-1)!}{x!(n-x-1)!} + \frac{(n-1)!}{(x-1)!(n-x)!} \\ &= (n-1)! \times \frac{(n-x)+x}{x!(n-x)!} = \binom{n}{x}.\end{aligned}$$

The first (second) term corresponds to the scenario where there are x $(x-1)$ successes out of $(n-1)$ trials and the last trial ends in a failure (success).

6.3.4 NORMAL DISTRIBUTION

As another important example of a continuous random variable, we introduce the *normal distribution*. This distribution is also called the *Gaussian distribution*, named after German mathematician Carl Friedrich Gauss. As implied by its name, the normal distribution is special because, as section 6.4.2 will explore, the sum of many random variables from the same distribution tends to follow the normal distribution even when the original distribution is not normal.

A normal random variable can take any number on the real line $(-\infty, \infty)$. The normal distribution has two parameters, mean μ and standard deviation σ. If X is a normal random variable, we may write $X \sim \mathcal{N}(\mu, \sigma^2)$, where σ^2 represents the variance (the square of standard deviation). The PDF and the CDF of the normal distribution are given by the following formulas.

The probability density function (PDF) of a **normal random variable** is given by

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\},$$

for any x on the real line. The cumulative probability distribution (CDF) has no analytically tractable form and is given by

$$F(x) = P(X \leq x) = \int_{-\infty}^{x} f(t)\,dt = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{1}{2\sigma^2}(t-\mu)^2\right\} dt, \tag{6.29}$$

where $X \sim \mathcal{N}(\mu, \sigma^2)$ and $\exp(\cdot)$ is the exponential function (see section 3.4.1). The CDF represents the area under the PDF from negative infinity up to x.

Figure 6.9 plots the PDF (left-hand plot) and CDF (right-hand plot) for the normal distribution, with three different sets of the mean and standard deviation. The PDF of the normal distribution is bell shaped and centered around its mean, with the standard deviation controlling the spread of the distribution. When the mean is 0 and standard deviation is 1, we have the *standard normal distribution*. The PDF is symmetric around the mean. Different means shift the PDF and CDF without changing their shape. In contrast, a larger standard deviation means more variability, yielding a flatter PDF and a more gradually increasing CDF.

The normal distribution has two important properties. First, adding a constant to (or subtracting it from) a normal random variable yields a normal random variable with appropriately shifted mean. Second, multiplying (or dividing) a normal random variable by a constant also yields another normal random variable with an appropriately scaled mean and standard deviation. Accordingly, the z-score of a normal random variable follows the standard normal distribution. We formally state these properties below.

Suppose X is a normal random variable with mean μ and standard deviation σ, i.e., $X \sim \mathcal{N}(\mu, \sigma^2)$. Let c be an arbitrary constant. Then, the following properties hold:

1. A random variable defined by $Z = X + c$ also follows a normal distribution, with $Z \sim \mathcal{N}(\mu + c, \sigma^2)$.
2. A random variable defined by $Z = cX$ also follows a normal distribution, with $Z \sim \mathcal{N}(c\mu, (c\sigma)^2)$.

These properties imply that the **z-score** of a normal random variable follows the standard normal distribution, which has zero mean and unit variance:

$$z\text{-score} = \frac{X - \mu}{\sigma} \sim \mathcal{N}(0, 1).$$

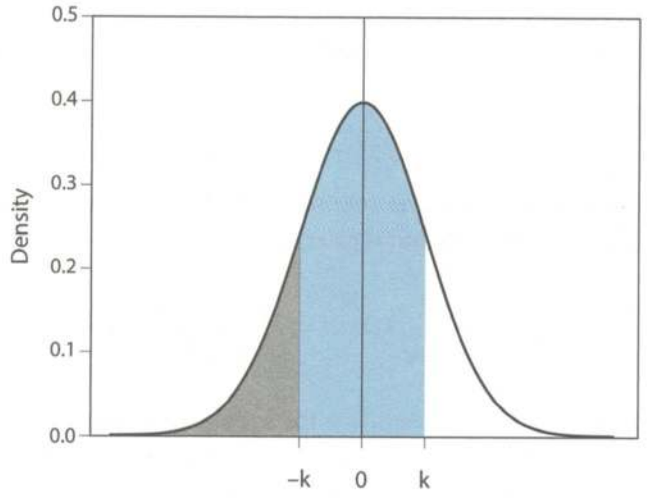


Figure 6.10. The Area under the Probability Density Function Curve of the Normal Distribution. The blue area can be computed as the difference between the cumulative distribution function (CDF) evaluated at k and $-k$ (i.e., the gray and blue areas minus the gray area).

In addition, it is important to note that if the data are distributed according to the normal distribution, about two-thirds are within 1 standard deviation from the mean and approximately 95% are within 2 standard deviations from the mean. Let us compute the probability that a normal random variable with mean μ and standard deviation σ lies within k standard deviations from the mean for a positive constant $k > 0$. To simplify the computation, consider the *z-score*, which has the standard normal distribution:

$$\begin{aligned} P(\mu - k\sigma \leq X \leq \mu + k\sigma) &= P\left(-k\sigma \leq X - \mu \leq k\sigma\right) \\ &= P\left(-k \leq \frac{X-\mu}{\sigma} \leq k\right) \\ &= P(-k \leq Z \leq k), \end{aligned}$$

where Z is a standard normal random variable. The first equality holds because we subtract μ from each term whereas the second inequality holds since we divide each term by a positive constant σ.

Thus, the desired probability equals the probability that a standard normal random variable lies between $-k$ and k. As illustrated in figure 6.10, this probability can be written as the difference in the CDF evaluated at k and $-k$:

$$P(-k \leq Z \leq k) = P\left(Z \leq k\right) - P\left(Z \leq -k\right) = F(k) - F(-k),$$

where $F(k)$ represents the sum of the blue and gray areas in the figure, whereas $F(-k)$ equals the gray area. These results can be confirmed in R with the `pnorm()` function, which evaluates the CDF at its input value. This function takes the mean (`mean`) and standard deviation (`sd`) as two important arguments. The default is the standard normal distribution with `mean = 0` and `sd = 1`.

```
## plus minus 1 standard deviation from the mean
pnorm(1) - pnorm(-1)

## [1] 0.6826895

## plus minus 2 standard deviations from the mean
pnorm(2) - pnorm(-2)

## [1] 0.9544997
```

The result suggests that, under the standard normal distribution, approximately 2/3 are within 1 standard deviation from the mean and about 95% are within 2 standard deviations from the mean. We can also directly specify mean and standard deviation without transforming a variable into a standard normal random variable. Suppose that the original distribution has a mean of 5 and standard deviation of 2, i.e., $\mu = 5$ and $\sigma = 2$. We can compute the same probabilities as above in the following way.

```
mu <- 5
sigma <- 2
## plus minus 1 standard deviation from the mean
pnorm(mu + sigma, mean = mu, sd = sigma) - pnorm(mu - sigma, mean = mu, sd = sigma)

## [1] 0.6826895

## plus minus 2 standard deviations from the mean
pnorm(mu + 2*sigma, mean = mu, sd = sigma) - pnorm(mu - 2*sigma, mean = mu, sd = sigma)

## [1] 0.9544997
```

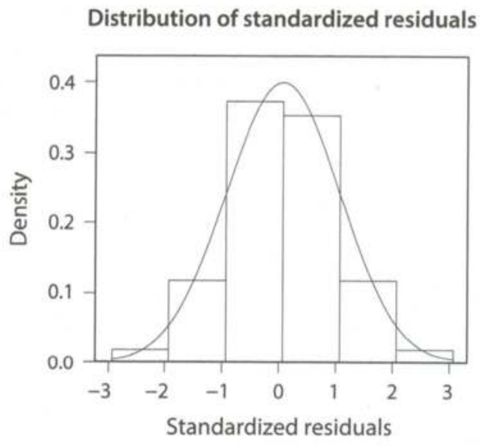
As an application of the normal distribution, consider the *regression towards the mean* phenomenon discussed in section 4.2.4. In that section, we presented evidence from US presidential elections demonstrating that in states where Obama received a large share of votes in 2008, he was likely to receive a *smaller* share of votes in 2012 (see section 4.2.5). Recall that our regression model used Obama's 2008 statewide vote share to predict his vote share for the same state in the 2012 election. We use the regression object `fit1`, as created in section 4.2.5.

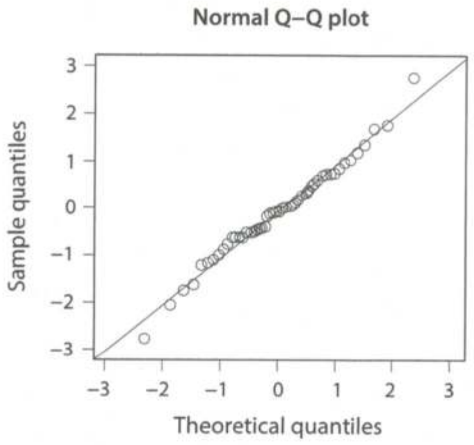
```
## see the page reference above
## "Obama2012.z" is Obama's 2012 standardized vote share
## "Obama2008.z" is Obama's 2008 standardized vote share
fit1

##
## Call:
## lm(formula = Obama2012.z ~ -1 + Obama2008.z, data = pres)
```

```
##
## Coefficients:
## Obama2008.z
##      0.9834
```

We examine the distribution of *residuals* and compare it with the normal distribution (see section 4.2.3 for the definition of residuals). We first present a histogram and overlay the PDF of the normal distribution using the `dnorm()` function. We then use a *quantile–quantile plot* (Q–Q *plot*) to directly compare the distribution of residuals with the normal distribution. The `qqnorm()` function creates a quantile–quantile plot using the *standard normal distribution*, whose mean is 0 and standard deviation is 1. To make the standard normal distribution and the distribution of residuals comparable, we use the `scale()` function to compute the *z-score* of residuals, or *standardized residuals*, whose mean is 0 and standard deviation is 1 (see section 3.7.3). Since residuals always have a mean of 0 (see section 4.2.3), we need only divide them by their standard deviation to obtain standardized residuals.

```
e <- resid(fit1)
## z-score of residuals
e.zscore <- scale(e)
## alternatively we can divide residuals by their standard deviation
e.zscore <- e / sd(e)
hist(e.zscore, freq = FALSE, ylim = c(0, 0.4),
     xlab = "Standardized residuals",
     main = "Distribution of standardized residuals")
x <- seq(from = -3, to = 3, by = 0.01)
lines(x, dnorm(x)) # overlay the normal density
qqnorm(e.zscore, xlim = c(-3, 3), ylim = c(-3, 3)) # quantile-quantile plot
abline(0, 1) # 45-degree line
```





Both the histogram and Q–Q plot show that the distribution of standardized residuals is remarkably close to the standard normal distribution. Now, consider the following probability model:

$$\text{Obama's 2012 standardized vote share} = 0.983 \times \text{Obama's 2008 standardized vote share} + \epsilon, \quad (6.30)$$

where 0.983 is the estimated slope coefficient, and the error term ϵ follows a normal distribution with mean and standard deviation equal to 0 and 0.18, respectively. The value of standard deviation is obtained as follows.

```
e.sd <- sd(e)
e.sd
## [1] 0.1812239
```

Thus, this probability model describes a potential data-generating process for Obama's 2012 vote share given his vote share in the previous election. Because both the outcome variable and the predictor are standardized, the intercept is estimated to be exactly zero and hence is not included in the `coef(fit1)` object (recall that the regression line always goes through the means of the outcome variable and the predictor).

We first analyze California where, in 2008, Obama won 61% of the votes, or a standardized vote share of 0.87. According to the above model, what is the probability that Obama wins a greater share of California votes in 2012? Using the `pnorm()` function, we can compute the area corresponding to the 2008 vote share under the normal distribution derived for Obama's 2012 votes from the probability model given in equation (6.30). We set the `lower.tail` argument in the `pnorm()` function to `FALSE` in order to compute the probability that Obama wins a *greater* vote share in 2012 than in 2008.

```
CA.2008 <- pres$Obama2008.z[pres$state == "CA"]
CA.2008
## [1] 0.8720631
CA.mean2012 <- coef(fit1) * CA.2008
CA.mean2012
## Obama2008.z
##   0.8576233
## area to the right; greater than CA.2008
pnorm(CA.2008, mean = CA.mean2012, sd = e.sd, lower.tail = FALSE)
## [1] 0.4682463
```

Thus, Obama is somewhat unlikely to win a larger share of California votes in 2012 than he won in 2008. In fact, the probability of this event is only 46.8%. Now consider Texas, where in 2008 Obama received only 44% of the votes, or a standardized vote share of −0.67. Again, under the probability model specified in equation (6.30), we compute the probability that Obama wins a greater share of Texas votes in 2012 than he did in the previous election.

```
TX.2008 <- pres$Obama2008.z[pres$state == "TX"]
TX.mean2012 <- coef(fit1) * TX.2008
TX.mean2012

## Obama2008.z
##  -0.6567543

pnorm(TX.2008, mean = TX.mean2012, sd = e.sd, lower.tail = FALSE)

## [1] 0.5243271
```

In the case of Texas, this probability is 52.4%, which is higher than the probability for California. This illustrates the regression towards the mean phenomenon under the probability model based on linear regression with a normally distributed error.

6.3.5 EXPECTATION AND VARIANCE

We have introduced several commonly used random variables by defining their PDF/PMF and CDF. These functions completely characterize the distribution of a random variable, but often it is helpful to obtain a more concise summary of a distribution. Previously, we used means and standard deviations in order to measure the center and spread of a distribution. We begin by examining the *expectation*, or mean, of a random variable. We should not confuse this with the *sample mean* discussed earlier in this book. The sample mean refers to the average of a variable in a particular data set, whereas the expectation or *population mean* represents the mean value under a probability distribution. The sample mean fluctuates from one sample to another, but the expectation of a random variable is of a theoretical nature and is fixed given a probability model.

Before we examine the formal definition of expectation, a few examples will prove instructive. Consider a Bernoulli random variable with success probability p (e.g., a single coin flip with the probability of landing on heads being p). What is the expectation? This random variable can take only two values, 0 (tail) and 1 (heads), and so the expectation can be computed as the weighted average of these two values with $(1-p)$ and p (i.e., the PMF) as weights, respectively. Let $\mathbb{E}(X)$ represent the expectation of a random variable X. Then, the expectation of a Bernoulli random variable can be computed as

$$\mathbb{E}(X) = 0 \times P(X=0) + 1 \times P(X=1) = 0 \times f(0) + 1 \times f(1) = 0 \times (1-p) + 1 \times p = p. \tag{6.31}$$

Similarly, consider a binomial random variable with success probability p and size n (e.g., the number of heads out of n independent and identical coin flips). This random variable can take any nonnegative integer up to n (i.e., $0, 1, \ldots, n$). The expectation of this binomial random variable is also defined as the weighted average of these values with the weights given by the corresponding values of the PMF:

$$\mathbb{E}(X) = 0 \times f(0) + 1 \times f(1) + \cdots + n \times f(n) = \sum_{x=0}^{n} x \times f(x). \tag{6.32}$$

While we use the weighted average to define expectation for a discrete random variable, we need a different way of defining the expectation for a continuous variable. We still compute the weighted average of each value in which the weights are given by the PDF. However, the difference is that a continuous random variable can take an uncountably infinite number of distinct values. This is done through the mathematical operation called *integration*. Readers who are not familiar with calculus can skip the details, but, for example, the expectation of a uniform random variable with interval $[a, b]$ is calculated as

$$\mathbb{E}(X) = \int_a^b x \times f(x)\,dx = \int_a^b \frac{x}{b-a}\,dx = \left.\frac{x^2}{2(b-a)}\right|_a^b = \frac{a+b}{2}. \tag{6.33}$$

Since each point within the interval is equally likely, the expectation of a uniform random variable equals the midpoint of the interval.

We now summarize the general definition of expectation for discrete and continuous random variables.

The **expectation** of a random variable is denoted by $\mathbb{E}(X)$ and is defined as

$$\mathbb{E}(X) = \begin{cases} \sum_x x \times f(x) & \text{if } X \text{ is discrete,} \\ \int x \times f(x)\,dx & \text{if } X \text{ is continuous,} \end{cases} \tag{6.34}$$

where $f(x)$ is the probability mass function or PMF (probability density function or PDF) of the discrete (continuous) random variable X.

In the definition of expectation, the summation and integration are taken with respect to all possible values of X. The set of all possible values that X takes is called the *support* of the distribution. We now introduce the basic rules of the expectation operator $\mathbb{E}$.

Let X and Y be random variables, and a and b be arbitrary constants. The **expectation** is a linear operator that satisfies the following equalities:

1. $\mathbb{E}(a) = a$.
2. $\mathbb{E}(aX) = a\mathbb{E}(X)$.
3. $\mathbb{E}(aX + b) = a\mathbb{E}(X) + b$.
4. $\mathbb{E}(aX + bY) = a\mathbb{E}(X) + b\mathbb{E}(Y)$.
5. If X and Y are independent, then $\mathbb{E}(XY) = \mathbb{E}(X)\mathbb{E}(Y)$. But generally, $\mathbb{E}(XY) \neq \mathbb{E}(X)\mathbb{E}(Y)$.

Now, using these rules, we can easily compute the expectation of a binomial random variable. Recall that a binomial random variable X with success probability p and size n is the sum of n independently and identically distributed (i.i.d.) Bernoulli random variables, $Y_1, \ldots, Y_n$, with the same success probability p. This suggests that we can obtain the expectation of the binomial random variable as

$$\mathbb{E}(X) = \mathbb{E}\left(\sum_{i=1}^{n} Y_i\right) = \sum_{i=1}^{n} \mathbb{E}(Y_i) = np.$$

This derivation is much more straightforward than the calculation that would be required (i.e., the sum of binomial PMFs evaluated at many values) if we used the definition of expectation given in equation (6.32).

Another useful statistic is the *standard deviation* and its square, *variance*, of a random variable. Both concepts have already been introduced in section 2.6.2. Like the expectation, it is important to distinguish between the standard deviation of a particular sample and the theoretical standard deviation of a random variable. Their interpretations match in that standard deviation is defined as the root mean square (RMS) of deviation from the mean (see section 2.6.2). In the current context, however, we use the expectation, rather than the sample average, to represent the mean.

The **variance** of a random variable X is defined as

$$\mathbb{V}(X) = \mathbb{E}[\{X - \mathbb{E}(X)\}^2].$$

The square root of $\mathbb{V}(X)$ is called the **standard deviation**.

Using the basic rules of expectation, we can write the variance as the difference between the expectation of X^2 and the expectation of X. The expectation of X^2 is called the *second moment*, while the expectation of X, or the mean, is called the *first moment*:

$$\begin{aligned}
\mathbb{V}(X) &= \mathbb{E}[\{X - \mathbb{E}(X)\}^2] \\
&= \mathbb{E}[X^2 - 2X\mathbb{E}(X) + \{\mathbb{E}(X)\}^2] \\
&= \mathbb{E}(X^2) - 2\mathbb{E}(X)\mathbb{E}(X) + \{\mathbb{E}(X)\}^2 \\
&= \mathbb{E}(X^2) - \{\mathbb{E}(X)\}^2. \qquad (6.35)
\end{aligned}$$

This alternative expression of variance is useful. For example, the variance of a Bernoulli random variable can be derived by noting that $X = X^2$ regardless of whether X equals 1 or 0 (because $1^2 = 1$ and $0^2 = 0$):

$$\mathbb{V}(X) = \mathbb{E}(X) - \{\mathbb{E}(X)\}^2 = p(1 - p). \tag{6.36}$$

This variance is greatest when $p = 0.5$. This makes intuitive sense because when p is smaller, for example, a Bernoulli random variable is more likely to equal 0 and hence has a smaller variance and hence less variation.

Similarly, using equation (6.35), we can also calculate the variance of a uniform random variable with the interval $[a, b]$, though readers unfamiliar with integration may ignore the details of the following derivation:

$$\begin{aligned}\mathbb{V}(X) &= \mathbb{E}(X^2) - \{\mathbb{E}(X)\}^2 = \int_a^b \frac{x^2}{b-a}\,dx - \left(\frac{a+b}{2}\right)^2 \\ &= \left.\frac{x^3}{3(b-a)}\right|_a^b - \left(\frac{a+b}{2}\right)^2 = \frac{1}{12}(b-a)^2.\end{aligned} \tag{6.37}$$

Like expectation, variance can be approximated through Monte Carlo simulation. Using the set of Bernoulli draws we generated earlier, we compute the sample variance, which should approximate the population variance.

```
## theoretical variance: p was set to 0.5 earlier
p * (1 - p)
## [1] 0.25
## sample variance using "y" generated earlier
var(y)
## [1] 0.2498088
```

Variance has several important properties. For example, since variance involves the expectation of squared distance from the mean, adding a constant to a random variable only shifts the variable and its mean by the same amount without altering its variance. However, the multiplication of a constant and a random variable changes its variance:

$$\mathbb{V}(aX) = \mathbb{E}[\{aX - a\mathbb{E}(X)\}^2] = a^2\mathbb{V}(X). \tag{6.38}$$

We summarize these properties below.

Let X and Y be random variables, and a and b be arbitrary constants. The **variance** operator $\mathbb{V}$ has the following properties:

1. $\mathbb{V}(a) = 0$.
2. $\mathbb{V}(aX) = a^2\mathbb{V}(X)$.
3. $\mathbb{V}(X + b) = \mathbb{V}(X)$.
4. $\mathbb{V}(aX + b) = a^2\mathbb{V}(X)$.
5. If X and Y are independent, $\mathbb{V}(X + Y) = \mathbb{V}(X) + \mathbb{V}(Y)$.

To compute the variance of a binomial random variable X, we use its status as the sum of n independently and identically distributed (i.i.d.) Bernoulli random variables, $Y_1, Y_2, \ldots, Y_n$, with success probability p:

$$\mathbb{V}(X) = \mathbb{V}\left(\sum_{i=1}^{n} Y_i\right) = \sum_{i=1}^{n} \mathbb{V}(Y_i) = np(1-p).$$

As another example, consider two independent normal random variables X and Y. Suppose that X has mean μ_X and variance σ_X^2, whereas Y has mean μ_Y and variance σ_Y^2. We write this setting compactly as $X \sim \mathcal{N}(\mu_X, \sigma_X^2)$ and $Y \sim \mathcal{N}(\mu_Y, \sigma_Y^2)$. What is the distribution of $Z = aX + bY + c$? The discussion in section 6.3.4 implies that Z is also a normal random variable. Using the rules of expectation and variance, we can derive the mean and variance as

$$\begin{aligned} \mathbb{E}(Z) &= a\mathbb{E}(X) + b\mathbb{E}(Y) + c = a\mu_X + b\mu_Y + c, \\ \mathbb{V}(Z) &= \mathbb{V}(aX + bY + c) = a^2\mathbb{V}(X) + b^2\mathbb{V}(Y) = a^2\sigma_X^2 + b^2\sigma_Y^2, \end{aligned}$$

respectively. Therefore, we have $Z \sim \mathcal{N}(a\mu_X + b\mu_Y + c, a^2\sigma_X^2 + b^2\sigma_Y^2)$.

6.3.6 PREDICTING ELECTION OUTCOMES WITH UNCERTAINTY

We next revisit the prediction of election outcomes using preelection polls. In section 4.1's introduction of the topic, our prediction did not include a measure of uncertainty. However, polling has *sampling variability* because we interview only a fraction of a large population. Suppose that we conduct a preelection poll under the exact same conditions multiple times. Each time, we obtain a representative sample of the target population and yet the sample consists of different voters. This means that the estimated support for a candidate will differ for each sample.

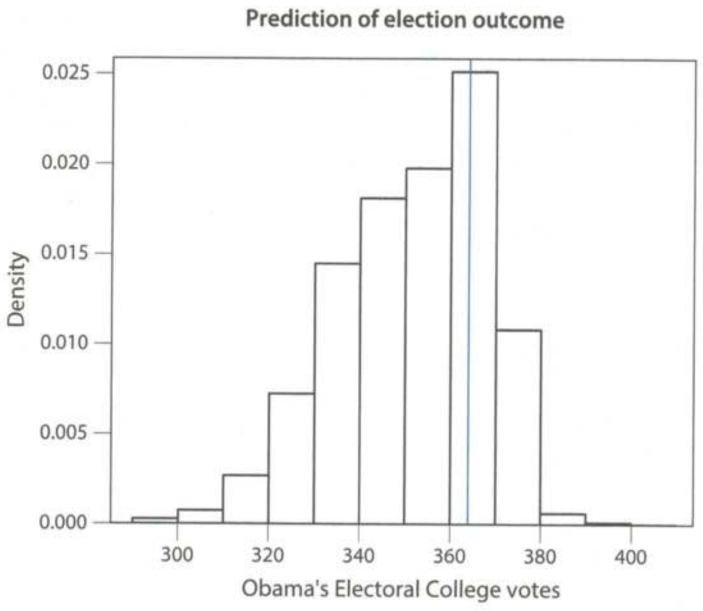
To capture this sampling variability, consider the following probability model. Suppose that the Election Day outcome represents the true proportion of Obama and McCain supporters in the population of voters within each state. We further assume that the fraction of voters who support a third-party candidate is negligible. We therefore focus on the two-party support rate for Obama, p_j, and McCain, $1 - p_j$, within each state j. The CSV data file, `pres08.csv`, contains the 2008 US presidential election results (see table 4.1). We first compute the two-party support rate for Obama.

```
pres08 <- read.csv("pres08.csv")
## two-party vote share
pres08$p <- pres08$Obama / (pres08$Obama + pres08$McCain)
```

We assume that for each hypothetical sampling, we interview 1000 voters who are randomly selected from the population. The binomial distribution with success probability p and size 1000 within each state is our model for Obama's support estimate based on a preelection poll. Using *Monte Carlo simulation*, we estimate Obama's support within each state, then allocate that state's Electoral College votes to the winning candidate. We will repeat this procedure many times to describe the uncertainty in preelection polling estimates that is due to sampling variability.

To sample from the binomial distribution in R, we use the `rbinom()` function. The `prob` argument of this function can take a vector of success probabilities. For each success probability, the function will return a vector of binomial random variable realizations. That is, given a p_j probability of success and $n = 1000$ voters, R will generate the number of votes for Obama. If a majority of these 1000 voters support Obama, we assign the state's Electoral College votes to Obama. We construct a histogram of these predicted Electoral College votes for Obama.

```
n.states <- nrow(pres08) # number of states
n <- 1000 # number of respondents
sims <- 10000 # number of simulations
## Obama's electoral votes
Obama.ev <- rep(NA, sims)
for (i in 1:sims) {
    ## samples number of votes for Obama in each state
    draws <- rbinom(n.states, size = n, prob = pres08$p)
    ## sums state's Electoral College votes if Obama wins the majority
    Obama.ev[i] <- sum(pres08$EV[draws > n / 2])
}
hist(Obama.ev, freq = FALSE, main = "Prediction of election outcome",
     xlab = "Obama's Electoral College votes")
abline(v = 364, col = "blue") # actual result
```



We find that all prediction draws are above the winning threshold of 270 votes. While the highest density of the histogram roughly corresponds to the actual number of Electoral College votes Obama won, the distribution of predictions is skewed. As a result, the mean and median values are lower than the actual number of Obama's votes.

```
summary(Obama.ev)
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   291.0   340.0   353.0   352.2   364.0   401.0
```

We can also analytically compute the expected value of Obama's Electoral College votes under this probability model. Let S_j represent the number of respondents (among a total of 1000 respondents) to a preelection poll who express support for Obama in state j. We use v_j to denote the number of Electoral College votes for state j. Then, the expected number of Obama's Electoral College votes is

$$\mathbb{E}(\text{Obama's votes}) = \sum_{j=1}^{51} v_j \times P(\text{Obama wins state } j) = \sum_{j=1}^{51} v_j \times P(S_j > 500). \tag{6.39}$$

To compute this expectation in R, we use the `pbinom()` function, which evaluates the CDF of the binomial distribution at its input value. As in `dbinom()`, the function takes as its arguments `size` and `prob`. In addition, we set the `lower.tail` argument to `FALSE` so that the function can be used to evaluate $P(S_j > 500)$ rather than $P(S_j \leq 500)$. The threshold 500 is based on the fact that we predict Obama as a winner for a state if more than half of 1000 respondents support him.

```
mean(Obama.ev)

## [1] 352.1646

## probability of binomial random variable taking greater than n/2 votes
sum(pres08$EV * pbinom(n / 2, size = n, prob = pres08$p, lower.tail = FALSE))

## [1] 352.1388
```

As expected, the analytically derived expected value is close to the approximate value based on Monte Carlo simulations. Similarly, we can compute the variance of Obama's electoral votes:

$$\begin{aligned}\mathbb{V}(\text{Obama's predicted votes}) &= \sum_{j=1}^{51} \mathbb{V}(v_j \mathbf{1}\{S_j > 500\}) \\ &= \sum_{j=1}^{51} v_j^2 P(S_j > 500)\big(1 - P(S_j > 500)\big).\end{aligned}$$

In this derivation, $\mathbf{1}\{\cdot\}$ represents the *indicator function*, which returns 1 (0) if the statement inside the curly braces is true (false). In addition, the first equality follows from the fact that the variance of the sum of independent random variables equals the sum of their respective variances. We also used the expression for the variance of a Bernoulli random variable given in equation (6.36) because we are evaluating the variance of a Bernoulli random variable $\mathbf{1}\{S_j > 500\}$. We compute the variance first with the theoretical expression above and then with Monte Carlo simulation draws.

```
## approximate variance using Monte Carlo draws
var(Obama.ev)

## [1] 268.7592

## theoretical variance
pres08$pb <- pbinom(n / 2, size = n, prob = pres08$p, lower.tail = FALSE)
V <- sum(pres08$pb * (1 - pres08$pb) * pres08$EV^2)
V

## [1] 268.8008

## approximate standard deviation using Monte Carlo draws
sd(Obama.ev)

## [1] 16.39388

## theoretical standard deviation
sqrt(V)

## [1] 16.39515
```

The result implies that with 1000 respondents in each state, our poll-based prediction of Obama's Electoral College votes varies from one sample to another. The standard deviation of our prediction is about 16 Electoral College votes. Given that Obama won the election with a much greater margin, this sampling variation did not significantly impact the preelection polls' ability to predict the winner.

6.4 Large Sample Theorems

As the final topic of this chapter, we introduce two important probabilistic regularities in large samples. In a wide range of probabilistic models, certain patterns will emerge as the sample size increases. These regularities will quantify the uncertainty of our data analysis in the next chapter. In this section, we discuss two *large sample theorems* (*asymptotic theorems*): the law of large numbers and the central limit theorem.

6.4.1 THE LAW OF LARGE NUMBERS

The *law of large numbers* states that as the sample size increases, the sample average converges to the expectation or population average.

Suppose that we obtain a random sample of n independently and identically distributed (i.i.d.) observations, $X_1, X_2, \ldots, X_n$, from a probability distribution with expectation $\mathbb{E}(X)$. The **law of large numbers** states

$$\overline{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i \rightarrow \mathbb{E}(X), \tag{6.40}$$

where we use $\rightarrow$ as shorthand for convergence.

In the theorem, X without subscript i represents a generic random variable, whereas X_i is the random variable for the ith observation. Although the precise mathematical meaning of convergence, as well as the precise conditions under which this theorem holds, are beyond the scope of this book, we emphasize that this theorem is applicable to a wide range of probability distributions. Intuitively speaking, the law states that the sample average, $\overline{X}_n$, will better approximate the expectation, $\mathbb{E}(X)$, as the sample size increases. The law of large numbers is powerful because it can be applied in most settings without knowledge of the underlying probability distribution.

We have already implicitly used the law of large numbers in a variety of contexts. The law of large numbers justifies the use of random sampling in surveys (see section 3.4.1). As we increase the number of randomly sampled respondents, the average response among them becomes closer to the true average of the population. In preelection polls, so long as the sample size is sufficiently large, the sample fraction of those who support Obama approximates the population fraction of voters who are Obama supporters. The law of large numbers enables researchers to talk to a small fraction of randomly sampled individuals in order to infer the opinion of the entire population.

In terms of a probability model, we can think of preelection polling as the sum of independently and identically distributed (i.i.d.) Bernoulli random variables, where a respondent is randomly drawn from a population of Obama supporters and non-supporters. That is, we define X_i as an indicator variable of voter i being an Obama supporter, i.e., $X_i = 1$ if voter i is an Obama supporter and $X_i = 0$ otherwise. The proportion of Obama supporters in the population is given by p. Then, the law of large numbers given in equation (6.40) can be directly applied. The sample fraction of Obama's supporters approaches the expectation, or the population proportion of Obama supporters, i.e., $\mathbb{E}(X) = p$.

Similarly, we can rely on the law of large numbers in randomized experiments when computing the difference-in-means between the (randomly divided) treatment and control groups to estimate the average treatment effect (see section 2.4.1). If we consider a population of potential outcomes, as the sizes of the treatment and control groups increase, the sample average of the observed outcome better approximates the expected potential outcome. In other words, we can apply the law of large numbers shown in equation (6.40) by setting X to each potential outcome, $Y(1)$ in the treatment group and $Y(0)$ in the control group.

The law of large numbers can also justify the use of *Monte Carlo simulations*. For example, in the birthday problem described in section 6.1.4, we computed the fraction of simulation trials where at least two birthdays were the same, in order to approximate the true probability of the event occurrence. When applying the law of large numbers shown in equation (6.40), this probability can be written as the expectation by defining a Bernoulli random variable that equals 1 if at least two birthdays match and 0 otherwise. We can then think of the fraction of simulation trials as the sample mean. Similarly, we solved the Monty Hall problem by computing the fraction of simulation trials in which a contender won a car rather than a goat (see section 6.2.2).

To illustrate the law of large numbers, we conduct a Monte Carlo simulation. We randomly sample from a binomial distribution with success probability $p = 0.2$ and size $n = 10$. We then examine, as the number of binomial draws increases, how the sample mean approaches the expectation, which equals $\mathbb{E}(X) = np = 2$ in this case. To calculate the sample mean after a single draw, two draws, and so on, all the way up to 1000 draws, we apply the `cumsum()` function. This function computes the *cumulative sum*, which combines all values up to and including the current value, for each position in a vector. For example, for a vector of length 3, (5, 3, 4), the `cumsum()` function will return another vector of length 3 that contains the cumulative sum (5, 8, 12). We obtain the desired average for each sample size (5, 4, 4) by dividing the cumulative sum vector by a vector that contains the number of elements used for the summation, i.e., (1, 2, 3). According to the law of large numbers, a large number of draws should produce a sample mean close to the expectation.

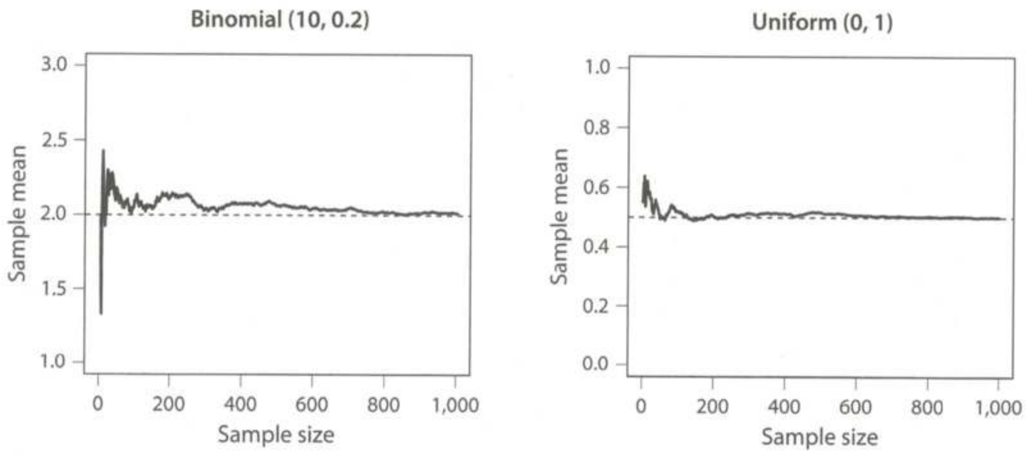
```
sims <- 1000
## 3 separate simulations for each
x.binom <- rbinom(sims, p = 0.2, size = 10)
## computing sample mean with varying sample size
mean.binom <- cumsum(x.binom) / 1:sims
```

In addition, we use the uniform distribution as an example of continuous random variables. The `runif()` function generates a random sample from this distribution.

```
## default runif() is uniform(0, 1)
x.unif <- runif(sims)
mean.unif <- cumsum(x.unif) / 1:sims
```

Finally, we plot the results. As the sample size increases, the sample mean approaches the expectation.

```
## plot for binomial
plot(1:sims, mean.binom, type = "l", ylim = c(1, 3),
     xlab = "Sample size", ylab = "Sample mean", main = "Binomial(10, 0.2)")
abline(h = 2, lty = "dashed") # expectation
## plot for uniform
plot(1:sims, mean.unif, type = "l", ylim = c(0, 1),
     xlab = "Sample size", ylab = "Sample mean", main = "Uniform(0, 1)")
abline(h = 0.5, lty = "dashed") # expectation
```



6.4.2 THE CENTRAL LIMIT THEOREM

The law of large numbers is useful but cannot quantify how good the approximation becomes as the sample size increases. For example, in the above figure, convergence appears to occur more quickly in the case of the uniform distribution than the binomial distribution. In practice, however, we observe only the sample mean and do not know the expectation. The former is something we compute from the data but the latter is a theoretical concept. Therefore, we need a different tool to know how well our sample mean approximates the expectation.

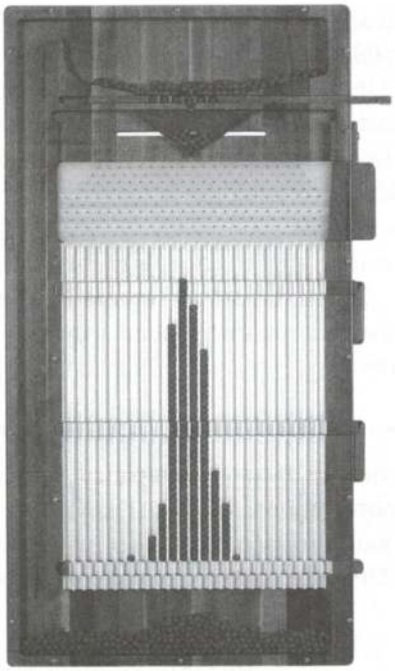


Figure 6.11. The Quincunx as a Machine to Illustrate the Central Limit Theorem.

The *central limit theorem* shows that the distribution of the sample mean approaches the *normal distribution* as the sample size increases. This is a remarkable result because, like the law of large numbers, it applies to a wide range of distributions. The result is useful, as shown in the next chapter, when quantifying the uncertainty of our estimates.

Before we explain the central limit theorem more formally, we discuss the *quincunx*, invented by Sir Francis Galton who first demonstrated the regression towards the mean phenomenon (section 4.2.4), as a machine that illustrates the theorem. Figure 6.11 presents a picture of a quincunx owned by the author. Red balls are dropped, one at a time, from the tiny hole at the top. The balls, as they fall, bounce off each peg either to its right or left before settling into one of the slots at the bottom of the machine. As seen in the figure, the balls will cluster in the middle, forming a bell-shaped curve that looks like a normal distribution.

Why does the quincunx create a bell-shaped curve? When a ball hits a peg, the ball has a 50–50 chance of bouncing off to its right or left. Although each path from the top to the bottom of the quincunx is equally likely, the ball has more ways to fall into a middle slot than a side slot. More formally, the total number of ways in which a ball reaches a particular slot can be computed using Pascal's triangle, as shown in figure 6.8. As illustrated in the figure, for example, if there are 5 lines of pegs in the quincunx, there are 20 ways for a ball to fall into the middle two slots.

We can understand the quincunx as a machine that generates a sequence of independently and identically distributed (i.i.d.) binomial random variables X with success probability 0.5 and size n, where n is the number of lines of pegs. Recall that a binomial random variable is the sum of n i.i.d. Bernoulli random variables. This means that if the central limit theorem holds, then we expect the binomial random variable to approximate the normal random variable as the sample size, or the number of lines of pegs in this case, increases. Here, the sample size refers to the number of lines of pegs, not the number of balls. Increasing the latter reduces the Monte Carlo error. In fact, this is exactly what we observe.

The central limit theorem applies not only to the Bernoulli random variable, but also to other distributions. This is important because in most practical settings we do not know the probability distribution that generates the data. We now more formally state the central limit theorem.

Suppose that we obtain a random sample of n independently and identically distributed (i.i.d.) observations, $X_1, X_2, \ldots, X_n$, from a probability distribution with mean $\mathbb{E}(X)$ and variance $\mathbb{V}(X)$. Let us denote the sample average by $\overline{X}_n = \sum_{i=1}^{n} X_i/n$. Then, the **central limit theorem** states

$$\frac{\overline{X}_n - \mathbb{E}(X)}{\sqrt{\mathbb{V}(X)/n}} \rightsquigarrow \mathcal{N}(0, 1). \tag{6.41}$$

In the theorem, $\rightsquigarrow$ indicates "convergence in distribution" as the sample size n increases.

While formula (6.41) appears complex at first glance, it has a straightforward interpretation. The theorem says that the *z-score* of the sample mean converges in distribution to the standard normal distribution or $\mathcal{N}(0, 1)$ as the sample size increases. Recall the definition of z-score given in equation (3.1). In order to standardize a random variable, we subtract its mean from it and then divide it by its standard deviation. As a result, any z-score has zero mean and unit variance.

To show that the left-hand side of formula (6.41) represents the z-score of the sample mean, we first note that the expectation of the sample mean $\overline{X}_n$ is the expectation of the original random variable X. Using the rules of the expectation operator, we obtain

$$\mathbb{E}(\overline{X}_n) = \mathbb{E}\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}(X_i) = \mathbb{E}(X). \tag{6.42}$$

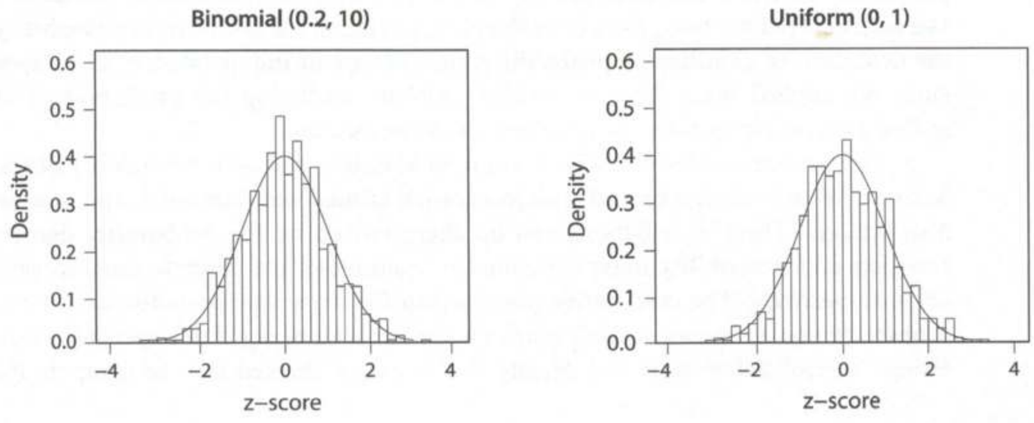
We next exploit the fact that the variance of two independent random variables equals the sum of their variances. The variance of the sample mean then is given by

$$\mathbb{V}(\overline{X}_n) = \mathbb{V}\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) = \frac{1}{n^2}\sum_{i=1}^{n}\mathbb{V}(X_i) = \frac{1}{n}\mathbb{V}(X). \tag{6.43}$$

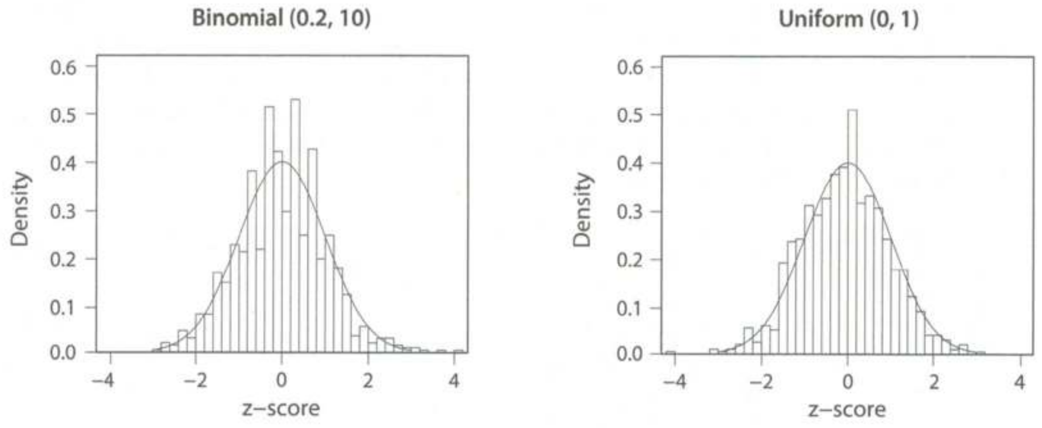
To derive this expression, we also used formula (6.38). This shows that the denominator of the left-hand side of formula (6.41) represents the standard deviation of the sample mean. Hence, the entire quantity in the left-hand side of formula (6.41) corresponds to the z-score of the sample mean.

Monte Carlo simulations can illustrate the central limit theorem. We consider two distributions as examples: the binomial distribution with success probability $p = 0.2$ and size $n = 10$, and the uniform distribution with the range $[0, 1]$. Recall that the mean and variance of this binomial distribution are $np = 10 \times 0.2 = 2$ and $np(1-p) = 10 \times 0.2 \times (1-0.2) = 1.6$, respectively. For this uniform distribution, the mean and variance are $(a+b)/2 = 1/2$ and $(b-a)^2/12 = 1/12$, respectively. We use these results to compute the z-scores and see whether their distributions can be approximated by the standard normal distribution. (To illustrate the quincunx through Monte Carlo simulations, we sample from the Bernoulli distribution or equivalently the Binomial distribution with size $n = 1$)

```
## sims = number of simulations
n.samp <- 1000
z.binom <- z.unif <- rep(NA, sims)
for (i in 1:sims) {
    x <- rbinom(n.samp, p = 0.2, size = 10)
    z.binom[i] <- (mean(x) - 2) / sqrt(1.6 / n.samp)
    x <- runif(n.samp, min = 0, max = 1)
    z.unif[i] <- (mean(x) - 0.5) / sqrt(1 / (12 * n.samp))
}
## histograms; nclass specifies the number of bins
hist(z.binom, freq = FALSE, nclass = 40, xlim = c(-4, 4), ylim = c(0, 0.6),
     xlab = "z-score", main = "Binomial(0.2, 10)")
x <- seq(from = -3, to = 3, by = 0.01)
lines(x, dnorm(x)) # overlay the standard normal PDF
hist(z.unif, freq = FALSE, nclass = 40, xlim = c(-4, 4), ylim = c(0, 0.6),
     xlab = "z-score", main = "Uniform(0, 1)")
lines(x, dnorm(x))
```



The above simulations are based on a sample size of 1000. We see that the standard normal distribution approximates the distribution of the z-score well. What about for a smaller sample size? Below, we conduct the same simulation using a sample size of 100 (the code is identical to the one above, aside from the change in sample size, and therefore omitted).



We observe that the approximation is poorer than before for the binomial distribution, whereas the central limit theorem holds well for the uniform distribution. The theorem does not tell us how large the sample size must be for a good approximation. As shown here, the answer to this question depends on the distribution of the original random variables. Nevertheless, what is incredible about the central limit theorem is that the z-score of the sample mean converges in distribution to the standard normal distribution *regardless of* the distribution of the original random variable.

6.5 Summary

In this chapter, we studied probability. We first introduced two different interpretations of probability, **frequentist** and **Bayesian**. Despite its competing interpretations, probability has a unified mathematical foundation with its basic definition and axioms. We then covered the basic rules of probability, including the **law of total probability**, the definition of **conditional probability**, the concept of **independence**, and **Bayes' rule**. We applied these rules to various problems including the prediction of an individual's race from their surname and residence location.

Next, we examined the concepts of **random variables** and their **probability distributions**. We introduced basic distributions such as **uniform**, **binomial**, and **normal** distributions. These distributions can be characterized by the **probability density function** and **probability mass function** for continuous and discrete random variables, respectively. The **cumulative distribution function** represents the cumulative probability that a random variable takes a value less than or equal to a specified value. Using the probability mass and density functions, we showed how to compute the

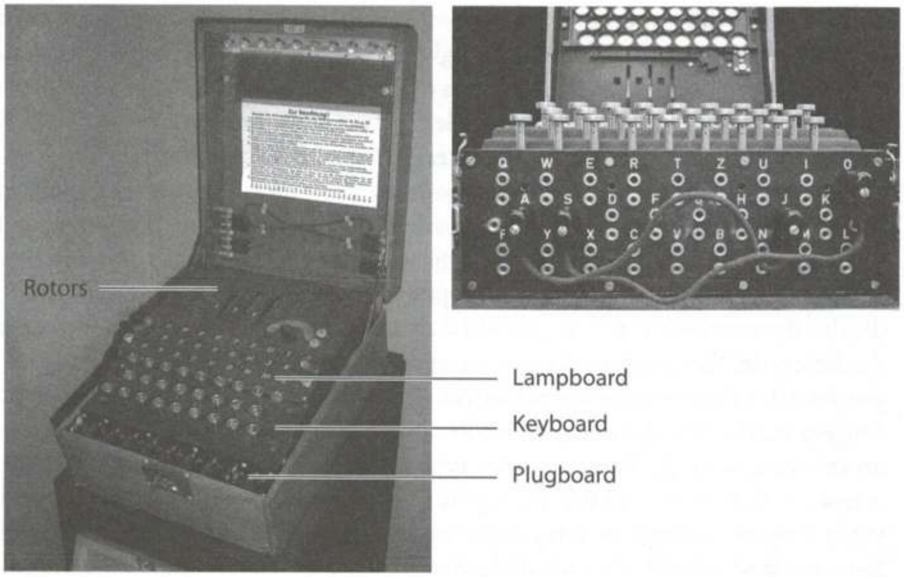


Figure 6.12. The Enigma Machine and its Plugboard. Photographer: Karsten Sperling, http://spiff.de/photo.

expectation and **variance** of a random variable. We used these tools to quantify the sampling uncertainty regarding the polling prediction of election results.

Lastly, we discussed the two fundamental large sample approximation theorems. The power of these theorems is that they can be applied to the sample mean of virtually any random variable given a sufficient sample size. The **law of large numbers** states that the sample mean approaches the expectation or the population mean as the sample size increases. This justifies the use of the sample mean as an estimator of the population mean in survey sampling and randomized experiments. The **central limit theorem** states that the z-score of the sample mean is approximately distributed according to the standard normal distribution. In the next chapter, we will use these large sample theorems to quantify the degree of uncertainty regarding the empirical conclusions drawn from our data analyses.

6.6 Exercises

6.6.1 THE MATHEMATICS OF ENIGMA

The Enigma machine is the most famous cipher machine to date. Nazi Germany used it during World War II to encrypt messages so that enemies could not understand them. The story of the British cryptanalysts who successfully deciphered Enigma has become the subject of multiple movies (*Enigma* (2001), *The Imitation Game* (2014)). In this exercise, we will focus our attention on a simplified version of the Enigma machine, which we name "Little Enigma." Like the real Enigma machine shown in the left panel of figure 6.12, this machine consists of two key components. First, the Little Enigma machine has 5 different *rotors*, each of which comes with 10 pins with numbers ranging from 0 to 9. Second, as shown in the right panel of figure 6.12, the *plugboard* contains 26 holes, corresponding to the 26 letters of the alphabet. In addition, 13 cables connect all possible pairs of letters. Since a cable has two ends, one can connect, for example, the letter A with any of the other 25 letters present in the plugboard.

To either encode a message or decode an encrypted message, one must provide the Little Enigma machine with the correct 5-digit passcode to align the rotors, and the correct configuration of the plugboard. The rotors are set up just like many combination locks. For example, the passcode 9–4–2–4–9 means that the 5 rotors display the numbers 9, 4, 2, 4, and 9 in that order. In addition, the 13 cables connecting the letters in the plugboard must be appropriately configured. The purpose of the plugboard is thus to scramble the letters. For example, if B is connected to W, the Little Enigma machine will switch B with W and W with B to encode a message or decode an encoded message. Thus, a sender types a message on the keyboard, the plugboard scrambles the letters, and the message is sent in its encrypted form. A receiver decodes the encrypted message by retyping it on a paired Little Enigma machine that has the same passcode and plugboard configuration.

1. How many different 5-digit passcodes can be set on the 5 rotors?

2. How many possible configurations does the plugboard provide? In other words, how many ways can 26 letters be divided into 13 pairs?

3. Based on the previous two questions, what is the total number of possible settings for the Little Enigma machine?

4. Five cryptanalytic machines have been developed to decode 1500 messages encrypted by the Little Enigma machine. The table below presents information on the number of messages assigned to each machine and the machine's failure rate (i.e., the percentage of messages the machine was unable to decode). Aside from this information, we do not know anything about the assignment of each message to a machine or whether the machine was able to correctly decode the message.

Machine	*Number of messages*	*Failure rate*
Banburismus	300	10%
Bombe	400	5%
Herivel tip	250	15%
Crib	340	17%
Hut 6	210	20%

Suppose that we select one message at random from the pool of all 1500 messages but find out this message was not properly decoded. Which machine is most likely responsible for this mistake?

5. Write an R function that randomly configures the plugboard. This function will take no input but will randomly select a set of 13 pairs of letters. The output object should be a 2×13 matrix for which each column represents a pair of letters. You may use the built-in R object `letters`, which contains the 26 letters of the alphabet as a character vector. Name the function `plugboard`.

6. Write an R function that encodes and decodes a message given a plugboard configuration set by the `plugboard()` function from the previous question. This function should take as inputs the output of the `plugboard()` function, as well as a message to be encoded (decoded), and return an encoded (decoded) message. You may wish to use the `gsub()` function, which replaces a pattern in a character string with another specified pattern. The `tolower()` function, which makes characters in a character vector lowercase, and `toupper()` function, which capitalizes characters in a character vector, can also help.

6.6.2 A PROBABILITY MODEL FOR BETTING MARKET ELECTION PREDICTION

Earlier in this chapter, we used preelection polls with a probability model to predict Obama's electoral vote share in the 2008 US election. In this exercise, we will apply a similar procedure to the Intrade betting market data analyzed in an exercise in chapter 4 (see section 4.5.1).[4] The 2008 Intrade data are available as `intrade08.csv`. The variable names and descriptions of this data set are available in table 4.9. Recall that each row of the data set represents daily trading information about the contracts for either the Democratic or Republican Party nominee's victory in a particular state. The 2008 election results data are available as `pres08.csv`, with variable names and descriptions appearing in table 4.1.

1. We analyze the contract of the Democratic Party nominee winning a given state j. Recall from section 4.5.1 that the data set contains the contract price of the market for each state on each day i leading up to the election. We will interpret `PriceD` as the probability p_{ij} that the Democrat would win state j if the election were held on day i. To treat `PriceD` as a probability, divide it by 100 so it ranges from 0 to 1. How accurate is this probability? Using only the data from the day before Election Day (November 4, 2008) within each state, compute the expected number of electoral votes Obama is predicted to win and compare it with the actual number of electoral votes Obama won. Briefly interpret the result. Recall that the actual total number of electoral votes for Obama is 365, not 364, which is the sum of electoral votes for Obama based on the results data. The total of 365 includes a single electoral vote that Obama garnered from Nebraska's 2nd Congressional District. McCain won Nebraska's 4 other electoral votes because he won the state overall.

[4] This exercise is based on David Rothschild (2009) "Forecasting elections: Comparing prediction markets, polls, and their biases." *Public Opinion Quarterly*, vol. 73, no. 5, pp. 895–916.

2. Next, using the same set of probabilities used in the previous question, simulate the total number of electoral votes Obama is predicted to win. Assume that the election in each state is a Bernoulli trial where the probability of success (Obama winning) is p_{ij}. Display the results using a histogram. Add the actual number of electoral votes Obama won as a solid line. Briefly interpret the result.

3. In prediction markets, people tend to exaggerate the likelihood that the trailing or "long shot" candidate will win. This means that candidates with a low (high) p_{ij} have a true probability that is lower (higher) than their predicted p_{ij}. Such a discrepancy could introduce bias into our predictions, so we want to adjust our probabilities to account for it. We do so by reducing the probability for candidates who have a less than 0.5 chance of winning, and increasing the probability for those with a greater than 0.5 chance. We will calculate a new probability p^*_{ij} using the following formula proposed by a researcher: $p^*_{ij} = \Phi(1.64 \times \Phi^{-1}(p_{ij}))$ where $\Phi(\cdot)$ is the CDF of a standard normal random variable and $\Phi^{-1}(\cdot)$ is its inverse, the quantile function. The R functions `pnorm()` and `qnorm()` can be used to compute $\Phi(\cdot)$ and $\Phi^{-1}(\cdot)$, respectively. Plot p_{ij}, used in the previous questions, against p^*_{ij}. In addition, plot this function itself as a line. Explain the nature of the transformation.

4. Using the new probabilities p^*_{ij}, repeat questions 1 and 2. Do the new probabilities improve predictive performance?

5. Compute the expected number of Obama's electoral votes using the new probabilities p^*_{ij} for each of the last 120 days of the campaign. Display the results as a time-series plot. Briefly interpret the plot.

6. For each of the last 120 days of the campaign, conduct a simulation as in question 2, using the new probabilities p^*_{ij}. Compute the quantiles of Obama's electoral votes at 2.5% and 97.5% for each day. Represent the range from 2.5% to 97.5% for each day as a vertical line, using a loop. Also, add the estimated total number of Obama's electoral votes across simulations. Briefly interpret the result.

6.6.3 ELECTION FRAUD IN RUSSIA

In this exercise, we use the rules of probability to detect election fraud by examining voting patterns in the 2011 Russian State Duma election.[5] The State Duma is the federal legislature of Russia. The ruling political party, United Russia, won this election, but to many accusations of election fraud, which the Kremlin, or Russian government, denied. As shown in figure 6.13, some protesters highlighted irregular patterns of voting as evidence of election fraud. In particular, the protesters pointed out the

[5] This exercise is based on Arturas Rozenas (2016) "Inferring election fraud from distributions of vote-proportions." Working paper.

Figure 6.13. Protesters in the Aftermath of the 2011 State Duma Election. The poster says, "We don't believe Churov! We believe Gauss!" Churov is the head of the Central Electoral Commission, and Gauss refers to an 18th century German mathematician, Carl Friedrich Gauss, whom the Gaussian (normal) distribution was named after. Source: Maxim Borisov, trv-science.ru.

Table 6.5. Russian and Canadian Election Data.

Variable	*Description*
`N`	total number of voters in a precinct
`turnout`	total turnout in a precinct
`votes`	total number of votes for the winner in a precinct

Note: The results of each election are stored in a data frame. The RData file `fraud.RData` contains data on four elections: the 2007 and 2011 Russian Duma elections, the 2012 Russian presidential election, and the 2011 Canadian election.

relatively high frequency of common fractions such as 1/4, 1/3, and 1/2 in the official vote shares.

We analyze the official election results, contained in the `russia2011` data frame in the RData file `fraud.RData`, to investigate whether there is any evidence for election fraud. The RData file can be loaded using the `load()` function. Besides `russia2011`, the RData file contains the election results from the 2003 Russian Duma election, the 2012 Russian presidential election, and the 2011 Canadian election, as separate data frames. Table 6.5 presents the names and descriptions of variables used in each data frame. **Note:** Part of this exercise may require computationally intensive code.

1. To analyze the 2011 Russian election results, first compute United Russia's vote share as a proportion of the voters who turned out. Identify the 10 most frequently occurring fractions for the vote share. Create a histogram that sets the number of bins to the number of unique fractions, with one bar created for each uniquely observed fraction, to differentiate between similar fractions like 1/2 and 51/100. This can be done by using the `breaks` argument in the `hist()` function. What does this histogram look like at fractions with low numerators and denominators such as 1/2 and 2/3?

2. The mere existence of high frequencies at low fractions may not imply election fraud. Indeed, more numbers are divisible by smaller integers like 2, 3, and 4 than by larger integers like 22, 23, and 24. To investigate the possibility that the low fractions arose by chance, assume the following probability model. The turnout for a precinct has a binomial distribution, whose size equals the number of voters and success probability equals the turnout rate for the precinct. The vote share for United Russia in this precinct is assumed to follow a binomial distribution, conditional on the turnout, where the size equals the number of voters who turned out and the success probability equals the observed vote share in the precinct. Conduct a Monte Carlo simulation under this alternative assumption (1000 simulations should be sufficient). What are the 10 most frequent vote share values? Create a histogram similar to the one in the previous question. Briefly comment on the results you obtain. **Note:** This question requires a computationally intensive code. Write a code with a small number of simulations first and then run the final code with 1000 simulations.

3. To judge the Monte Carlo simulation results against the actual results of the 2011 Russian election, we compare the observed fraction of observations within a bin of certain size with its simulated counterpart. To do this, create histograms showing the distribution of question 2's four most frequently occurring fractions, i.e., 1/2, 1/3, 3/5, and 2/3, and compare them with the corresponding fractions' proportion in the actual election. Briefly interpret the results.

4. We now compare the relative frequency of observed fractions with the simulated ones beyond the four fractions examined in the previous question. To do this, we choose a bin size of 0.01 and compute the proportion of observations that fall into each bin. We then examine whether or not the observed proportion falls within the 2.5 and 97.5 percentiles of the corresponding simulated proportions. Plot the result with the horizontal axis as the vote share and vertical axis as the estimated proportion. This plot will attempt to reproduce the one held by protesters in figure 6.13. Also, count the number of times that the observed proportions fall outside the corresponding range of simulated proportions. Interpret the results.

5. To put the results of the previous question into perspective, apply the procedure developed in the previous question to the 2011 Canadian elections and the 2003 Russian election, where no major voting irregularities were reported. In addition, apply this procedure to the 2012 Russian presidential election, where election fraud allegations were reported. No plot needs to be produced. Briefly comment on the results you obtain. **Note:** This question requires a computationally intensive code. Write a code with a small number of simulations first and then run the final code with 1000 simulations.

Chapter 7

Uncertainty

As far as the laws of mathematics refer to reality, they are not certain; and as far as they are certain, they do not refer to reality.
—Albert Einstein, *Geometry and Experience*

Thus far, we have studied various data analysis techniques that can extract useful information from data. We have used these methods to draw causal inferences, measure quantities of interest, make predictions, and discover patterns in data. One important remaining question, however, is how certain we can be of our empirical findings. For example, if in a randomized controlled trial the average outcome differs between the treatment and control groups, when is this difference large enough for us to conclude that the treatment of interest affects the outcome, on average? Did the observed difference result from chance? In this chapter, we consider how to separate signals from noise in data by quantifying the degree of uncertainty. We do so by applying the laws of probability introduced in the previous chapter. We cover several concepts and methodologies to formally quantify the level of uncertainty. These include bias, standard errors, confidence intervals, and hypothesis testing. Finally, we describe ways to make inferences from linear regression models with measures of uncertainty.

7.1 Estimation

In earlier chapters, we showed how to infer public opinion in a population through survey sampling (chapter 3) and estimate causal effects through randomized controlled trials (chapter 2). In these examples, researchers want to estimate the unknown value of a quantity of interest using observed data. We refer to the quantity of interest as a *parameter* and the method to compute its estimate as an *estimator*. For example, in the analysis of survey data presented in chapter 3, we are interested in estimating the proportion of Obama supporters in the population of American voters (parameter) based on a relatively small number of survey respondents (data). We use the sample proportion of Obama supporters as our estimator. Similarly, in randomized controlled trials, the average outcome difference between the treatment and control groups represents an estimator for the average causal effect, which is our parameter.

How good is our estimate of the parameter? This is a difficult question to answer because we do not know the true value of the parameter. However, it turns out that we can characterize how well the estimator will perform over hypothetically repeated sampling. This section shows how statistical theory can help us investigate the performance of the estimators we used in the earlier parts of the book.

7.1.1 UNBIASEDNESS AND CONSISTENCY

Consider a survey for which a certain number of respondents are selected from a population using the *simple random sampling* procedure. Simple random sampling implies that each individual in the population is equally likely to be selected into a sample. As discussed in chapter 3, such random sampling benefits us by producing a representative sample of a target population (see section 3.4.1).

To give further context to these ideas, recall the preelection polling example in the 2008 US presidential election (see section 4.1.3). In that example, our parameter was the proportion of voters in the population of American voters that supported Obama. We used simple random sampling to obtain a representative sample of n voters from the population. The survey asked whether each of the respondents supported Obama or not. We used the sample proportion of those who supported Obama as our estimate of the population proportion of Obama supporters.

To formalize the content of the previous paragraph, let p denote the population proportion of Obama supporters. We use a random variable X to represent a response to the question. If voter i supports (does not support) Obama, then we denote this observation with $X_i = 1$ ($X_i = 0$). Since each respondent is sampled independently from the same population, we can assume that $\{X_1, X_2, \ldots, X_n\}$ are independently and identically distributed (i.i.d.) Bernoulli random variables with success probability p (see section 6.3.2). Our estimator is the sample proportion, $\overline{X}_n = \sum_{i=1}^{n} X_i / n$, which we use to estimate the unknown parameter p. The specific value of this estimator we obtain from our sample represents the estimate of p.

How good is this estimate? Ideally, we would like to compute the *estimation error*, which is defined as the difference between our estimate and the truth:

$$\text{estimation error} = \text{estimate} - \text{truth} = \overline{X}_n - p.$$

However, the estimation error can never be computed because we do not know p. In fact, if we know the truth, there is no need to estimate the parameter in the first place!

While we never know the size of the estimation error specific to our sample, it is sometimes possible to compute the *average* magnitude of the estimation error. To do this, we consider the hypothetical scenario of conducting the same preelection poll infinitely many times in exactly the same manner. This scenario is purely hypothetical because in reality we obtain only one sample and can never conduct sampling in an identical manner multiple times. Under this scenario, each hypothetical poll would draw a different set of n voters from the sample population and yield a different proportion of sampled voters who express support for Obama. This means that the sample proportion, represented by a random variable $\overline{X}_n$, would take a different value for each poll. As a result, the estimation error would also differ from one poll to another and hence is a random variable.

More formally, the sample proportion can be considered as a random variable that has its own distribution over the repeated use of simple random sampling. This distribution is called the *sampling distribution* of the estimator. In this particular example, each hypothetical sample is drawn independently from the same population. Therefore, the sample proportion, $\overline{X}_n$, is a binomial random variable, divided by n, with success probability p and size n where n represents the number of respondents in a poll (recall from section 6.3.3 that the sum of i.i.d. Bernoulli random variables is a binomial random variable).

We now compute the average estimation error or *bias* over this repeated simple random sampling procedure using the concept of *expectation* (see section 6.3.5). Under the binomial model, the success probability equals p. Therefore, we can show that the bias, or the average estimation error, of the sample mean is zero:

$$\text{bias} = \mathbb{E}(\text{estimation error}) = \mathbb{E}(\text{estimate} - \text{truth}) = \mathbb{E}(\overline{X}_n) - p = p - p = 0.$$

This result implies that the sample proportion under simple random sampling is an *unbiased* estimator for the population proportion. That is, while the sample proportion based on a specific sample may deviate from the population proportion, it gives, on average, the right answer. More precisely, if we were to conduct the same preelection poll infinitely many times under identical conditions, the average of the sample proportions of Obama supporters would exactly equal their population proportion. Thus, unbiasedness refers to the accuracy of the average estimate over repeated sampling rather than the accuracy of an estimate based on the observed data.

Similar logic applies to nonbinary variables. We can show that the expectation of the sample mean equals the population average so long as each survey respondent is randomly sampled from a large population. An example of a nonindependent sampling procedure is respondent-driven sampling, in which one respondent introduces another respondent to the interviewer. Using the fact that expectation is a linear operator (see section 6.3.5), we obtain the following general result for the sample mean:

$$\mathbb{E}(\overline{X}_n) = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}(X_i) = \mathbb{E}(X). \tag{7.1}$$

The final equality follows because each of the n observations is randomly sampled from the same population whose mean is denoted by $\mathbb{E}(X)$. Therefore, regardless of the distribution of a variable, random sampling provides a way to use the sample average as an unbiased estimator of the population mean. In other words, equation (7.1) shows that random sampling eliminates bias.

In general, random sampling plays an essential role in obtaining an unbiased estimate. In the absence of random sampling or other ways to obtain a representative sample, it is difficult to estimate a population characteristic without bias. For example, item and unit *nonresponse*, discussed in section 3.4.2, can yield biased estimates.

In section 6.4.1, we introduced the *law of large numbers*, which states that as sample size increases, the sample mean converges to the population mean. In the current context, this implies that the estimation error, which is the difference between the sample mean and the population mean, becomes smaller as the sample size increases. The estimator is said to be *consistent* if it converges to the parameter as the sample size goes to infinity. Thus, the discussion so far implies that the sample mean is a good estimator for the population mean because it is an unbiased and consistent estimator of the population mean. That is, the sample mean on average correctly estimates the population mean, and the estimation error decreases as the sample size increases.

An estimator is said to be **unbiased** if its expectation equals the parameter. An estimator is said to be **consistent** if it converges to the parameter as the sample size increases. For example, the sample average $\overline{X}_n = \sum_{i=1}^{n} X_i / n$ is unbiased and consistent for the population mean $\mathbb{E}(X)$ under simple random sampling:

$$\mathbb{E}(\overline{X}_n) = \mathbb{E}(X) \quad \text{and} \quad \overline{X}_n \rightarrow \mathbb{E}(X).$$

We next show that the difference-in-means estimator used to analyze *randomized controlled trials* (see section 2.4) is unbiased for the average treatment effect. Suppose that we have a sample of n units for which we conduct a randomized experiment. This experiment features a single binary treatment T_i which equals 1 if unit i receives the treatment and 0 if the unit is assigned to the control group. We randomly choose n_1 units out of this sample and assign them to the treatment group, and the remaining $n - n_1$ units belong to the control group. This treatment assignment procedure is called *complete randomization*, which fixes a priori the total number of units that receive the treatment. In contrast, *simple randomization* randomly assigns treatment to each unit independently, and so the total number of treated units will vary from one randomization to another. Thus, under complete randomization, there exists a total of $\binom{n}{n_1}$ ways of assigning n_1 units to the treatment group and the remaining units to the control group (see section 6.1.5 for the definition of combinations). Each of these treatment assignment combinations is equally likely but only one of them is realized.

The first parameter we consider, the *sample average treatment effect* (SATE), is defined in equation (2.1) and reproduced here:

$$\text{SATE} = \frac{1}{n} \sum_{i=1}^{n} \{Y_i(1) - Y_i(0)\}.$$

In this equation, $Y_i(1)$ and $Y_i(0)$ are the potential outcomes under the treatment and control conditions for unit i, respectively. As discussed in section 2.3, $Y_i(1)$ ($Y_i(0)$) represents the outcome that would be observed for unit i if it were assigned to the treatment (control) condition. Since $Y_i(1) - Y_i(0)$ represents the treatment effect for unit i, the SATE is the average of this treatment effect across all units in the sample. But because only one potential outcome can be observed for each unit, we cannot observe the treatment effect for any unit, so the SATE is unknown.

In section 2.4, we learned that randomization of treatment assignment makes the treatment and control groups identical on average. As a result, we can use the *difference-in-means estimator* to estimate average treatment effect. Let's formalize this argument here. The difference-in-means estimator $\widehat{\mathsf{SATE}}$ can be written as

$$\begin{aligned}\widehat{\mathsf{SATE}} &= \text{average of the treated} - \text{average of the untreated} \\ &= \frac{1}{n_1}\sum_{i=1}^{n} T_i Y_i - \frac{1}{n-n_1}\sum_{i=1}^{n}(1-T_i)Y_i. \end{aligned} \tag{7.2}$$

Recall that n_1 represents the number of units in the treatment group and hence $n - n_1$ is the size of the control group. The expression $\sum_{i=1}^{n} T_i Y_i$, for example, gives the sum of the observed outcome variable across all treated units because the treatment variable T_i is 1 when unit i is treated and 0 if it belongs to the control group. This means that $T_i Y_i = Y_i$ and $(1 - T_i)Y_i = 0$ when observation i is in the treatment group, and $T_i Y_i = 0$ and $(1 - T_i)Y_i = Y_i$ when it is in the control group.

We now show that the difference-in-means estimator is unbiased for the SATE. As discussed earlier, in survey sampling, the unbiasedness of an estimator means that over repeated sampling the average value of the estimator is identical to the unknown true value of the parameter. In randomized controlled trials, we consider how an estimator behaves over the repeated randomization of treatment assignment. That is, suppose that using a sample of the same n units, a researcher conducts a randomized control trial (infinitely) many times by randomizing the treatment assignment. A given unit will receive the treatment in some of these trials while in others it will be assigned to the control group. Each time, a researcher will compute the difference-in-means estimator after randomizing the treatment assignment and observing the outcome. Throughout the hypothetical repeated experiments, the potential outcomes remain fixed and only the treatment assignment changes. Thus, unbiasedness implies that the average value of the difference-in-means estimator over repeated trials is equal to the true value of the SATE.

To show the unbiasedness more formally, we can take the expectation of the difference-in-means estimator with respect to T_i since in this framework the randomized treatment assignment T_i is the only random variable. Since T_i is a Bernoulli random variable, its expectation equals $P(T_i = 1)$, which is the proportion of subjects who are treated, or n_1/n in this case:

$$\begin{aligned}\mathbb{E}(\widehat{\mathsf{SATE}}) &= \mathbb{E}\left(\frac{1}{n_1}\sum_{i=1}^{n} T_i Y_i(1) - \frac{1}{n-n_1}\sum_{i=1}^{n}(1-T_i)Y_i(0)\right) \\ &= \frac{1}{n_1}\sum_{i=1}^{n}\mathbb{E}(T_i)Y_i(1) - \frac{1}{n-n_1}\sum_{i=1}^{n}\mathbb{E}(1-T_i)Y_i(0) \\ &= \frac{1}{n_1}\sum_{i=1}^{n}\frac{n_1}{n}Y_i(1) - \frac{1}{n-n_1}\sum_{i=1}^{n}\left(1-\frac{n_1}{n}\right)Y_i(0) \\ &= \frac{1}{n}\sum_{i=1}^{n}\{Y_i(1) - Y_i(0)\} = \mathsf{SATE}. \end{aligned} \tag{7.3}$$

The first equality follows because for a treated unit, the potential outcome under the treatment condition is observed, i.e., $Y_i = Y_i(1)$, while a control unit reveals the other potential outcome, i.e., $Y_i = Y_i(0)$. The second equality holds because the expectation is a linear operator and is taken with respect to the treatment assignment. That is, the potential outcomes are treated as fixed constants. The derivation above shows that the difference-in-means estimator is unbiased for the SATE.

We can combine the advantage of the above random treatment assignment with that of random sampling. Suppose that we first randomly sample n individuals from a large population of interest. Within this sample, we randomly assign the treatment to n_1 individuals and measure the outcome for each one. This two-step procedure ensures the experimental results are generalizable to the population because the experiment's sample is representative of the population. To see this formally, consider the *population average treatment effect* or PATE, which represents the average of the treatment effect among all individuals in the population. Here, we use the expectation to represent the population average:

$$\mathsf{PATE} = \mathbb{E}(Y(1) - Y(0)). \tag{7.4}$$

Recall that the sample is representative of the population because of random sampling. This means that while the SATE is unobservable, its expectation equals the PATE. Since the difference-in-means estimator is unbiased for the SATE, the estimator is also unbiased for the PATE. It is also clear from equation (7.3) that the difference-in-means estimator is consistent for the PATE. This result emerges from applying the *law of large numbers* to the sample average of the treatment group and that of the control group, separately. In sum, the combination of random sampling and random assignment enables us to make causal inferences about a target population.

In randomized controlled trials, the average outcome difference between the treatment and control groups is an unbiased estimator of the **sample average treatment effect** (SATE). The estimator is also unbiased and consistent for the **population average treatment effect** (PATE).

A *Monte Carlo simulation* can illustrate the idea of unbiasedness. Suppose that the potential outcome under the control condition $Y_i(0)$ is distributed according to the *standard normal distribution* in a population (i.e., a normal distribution with zero mean and unit variance). We further assume that in the population the individual-level treatment effect follows another normal distribution with both mean and variance equal to 1. Formally, we can write this hypothetical *data-generating process* as

$$Y_i(0) \sim \mathcal{N}(0, 1) \quad \text{and} \quad Y_i(1) \sim \mathcal{N}(1, 1). \tag{7.5}$$

The treatment assignment is randomized, where a randomly selected half of the sample receives the treatment and the other half does not. Finally, we can define the treatment effect for unit i as $\tau_i = Y_i(1) - Y_i(0)$. For each unit, we observe the potential outcome under the realized treatment condition. Under this model, we can analytically compute

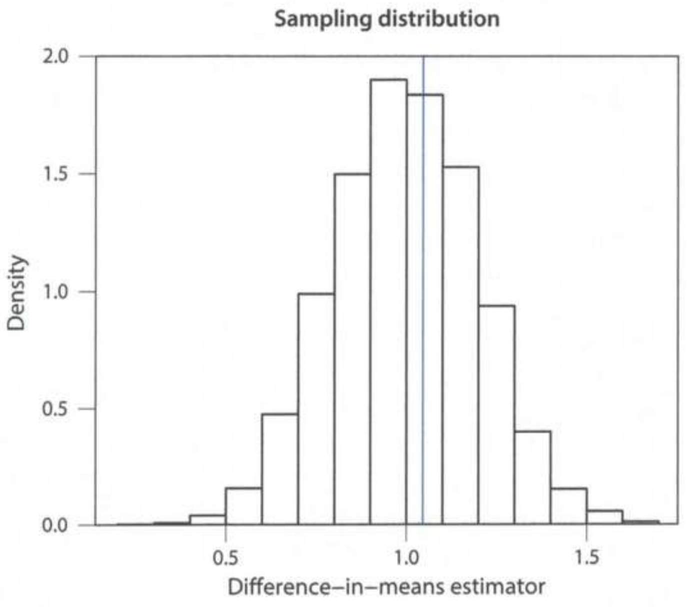
```
## unbiased
summary(est.error)

##       Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.757900 -0.140900 -0.003669 -0.002793  0.134400  0.650100
```

The average estimation error is close to zero, reflecting the unbiasedness. The variability is greater than in the case of the SATE because random sampling adds more noise.

7.1.2 STANDARD ERROR

We have focused on the mean of the estimation error, but an unbiased estimator with a large degree of variability is of little use in practice. In the above simulation example, the difference-in-means estimator was unbiased but its estimation error was sometimes large. We can plot the sampling distribution of the difference-in-means estimator. The histogram shows that while the estimator is accurate on average, it varies significantly from one randomized treatment assignment to another.

```
hist(diff.means, freq = FALSE,xlab = "Difference-in-means estimator",
     main = "Sampling distribution")
abline(v = SATE, col = "blue") # true value of SATE
```



How much would an estimator vary over the hypothetically repeated data-generating process? We have used the standard deviation to characterize the spread of distribution in earlier parts of the book, and we can do the same here. In the above simulation example, this amounts to calculating the standard deviation of the sampling distribution of the difference-in-means estimator.

```
sd(diff.means)

## [1] 0.2003772
```

The result implies that in this example, the difference-in-means estimator is on average 0.2 points away from its mean. This mean equals the true value of the SATE, since the difference-in-means estimator is unbiased for the SATE. Accordingly, the mean of the sampling distribution equals the true value of the SATE, implying that the standard deviation of the sampling distribution (i.e., the deviation from the mean) in this case is equal to the *root-mean-squared error* (RMSE; i.e., the deviation from the truth) (see section 4.1.3 for the definition of RMSE). In our simulation example, we can compute the RMSE as follows.

```
sqrt(mean((diff.means - SATE)^2))

## [1] 0.2062641
```

The result implies that the estimator is on average 0.206 points away from the true value of the SATE. The small difference between the standard deviation and the RMSE reflects the Monte Carlo error in which the sample average differs from its expectation by a small amount.

However, if an estimator is biased, then the standard deviation of its sampling distribution will differ from the RMSE. Formally, we can show that the *mean-squared error* (MSE), which is the square of the RMSE, equals the sum of the variance and squared bias. Let θ be a parameter and $\hat{\theta}$ be its estimator. We can derive this decomposition as follows:

$$\begin{aligned}
\text{MSE} &= \mathbb{E}\{(\hat{\theta}-\theta)^2\} \\
&= \mathbb{E}[\{(\hat{\theta}-\mathbb{E}(\hat{\theta}))+(\mathbb{E}(\hat{\theta})-\theta)\}^2] \\
&= \mathbb{E}[\{\hat{\theta}-\mathbb{E}(\hat{\theta})\}^2]+\{\mathbb{E}(\hat{\theta})-\theta\}^2 \\
&= \text{variance} + \text{bias}^2.
\end{aligned}$$

The second equality follows because we simply added and subtracted $\mathbb{E}(\hat{\theta})$. The third equality is based on the fact that the cross-product term obtained by expanding the square, i.e., $2\mathbb{E}\{(\hat{\theta}-\mathbb{E}(\hat{\theta}))(\mathbb{E}(\hat{\theta})-\theta)\}$, can be shown to equal zero.[1]

The decomposition implies that when assessing the accuracy of an estimator, we care about variance as well as bias. An unbiased estimator can have a greater MSE than a biased estimator if the variance of the former is sufficiently larger than that of the latter.

The above discussion suggests that we can characterize the variability of an estimator by computing the standard deviation of the *sampling distribution*. Unfortunately, this

[1] Specifically, using the rules of expectation, we have $\mathbb{E}\{(\hat{\theta}-\mathbb{E}(\hat{\theta}))(\mathbb{E}(\hat{\theta})-\theta)\} = \mathbb{E}[\hat{\theta}\mathbb{E}(\hat{\theta})-\hat{\theta}\theta-\{\mathbb{E}(\hat{\theta})\}^2+\mathbb{E}(\hat{\theta})\theta] = \{\mathbb{E}(\hat{\theta})\}^2-\mathbb{E}(\hat{\theta})\theta-\{\mathbb{E}(\hat{\theta})\}^2+\mathbb{E}(\hat{\theta})\theta = 0$.

The middle probability term, $\{\mathbb{E}(X) - \overline{X}_n\}/\text{standard error}$, equals the negative z-score of the sample mean, which has the same sampling distribution as the z-score because of symmetry. The central limit theorem implies that the z-score of the sample mean follows the standard normal distribution when the sample size is sufficiently large:

$$\frac{\overline{X}_n - \mathbb{E}(X)}{\sqrt{\mathbb{V}(X)/n}} \approx \frac{\overline{X}_n - \mathbb{E}(X)}{\text{standard error}} \sim \mathcal{N}(0, 1). \tag{7.15}$$

Therefore, the probability in equation (7.14) equals the blue area of figure 7.1.

We now summarize the standard procedure for constructing asymptotic confidence intervals based on the central limit theorem. The procedure applies to any estimator so long as its asymptotic sampling distribution can be approximated by the normal distribution. Such a normal approximation holds for many cases of interest including almost all the examples in this book.

The **confidence interval** of an estimate $\hat{\theta}$ can be obtained by using the following procedure:

1. Choose the desired level of confidence $(1 - \alpha) \times 100\%$ by specifying a value of α between 0 and 1: the most common choice is $\alpha = 0.05$, which gives a 95% confidence level.
2. Derive the sampling distribution of the estimator by computing its mean and variance: in the case of the sample mean, this is given by equation (7.12).
3. Compute the standard error based on this sampling distribution.
4. Compute the critical value $z_{\alpha/2}$ as the $(1 - \alpha/2) \times 100$ percentile value of the standard normal distribution: see table 7.1.
5. Compute the lower and upper confidence limits as $\hat{\theta} - z_{\alpha/2} \times \text{standard error}$ and $\hat{\theta} + z_{\alpha/2} \times \text{standard error}$, respectively.

The resulting confidence interval covers the true parameter value θ over a hypothetically repeated data-generating process $(1 - \alpha) \times 100\%$ of the time.

Several applications of this procedure will be given throughout this section. Here, we conduct Monte Carlo simulations to further illustrate the idea of confidence intervals. First we revisit the PATE simulation shown in section 7.1.2. Given the estimates and standard errors we computed, we can obtain the 90% and 95% confidence intervals for each of the 5000 simulations.

```
## empty container matrices for 2 sets of confidence intervals
ci95 <- ci90 <- matrix(NA, ncol = 2, nrow = sims)
## 95% confidence intervals
ci95[, 1] <- diff.means - qnorm(0.975) * se # lower limit
ci95[, 2] <- diff.means + qnorm(0.975) * se # upper limit
```

```
## 90% confidence intervals
ci90[, 1] <- diff.means - qnorm(0.95) * se # lower limit
ci90[, 2] <- diff.means + qnorm(0.95) * se # upper limit
```

If these confidence intervals are valid, then they should contain the true value of the PATE, which is equal to 1 in this simulation, approximately 95% and 90% of time, respectively. That is exactly what we find below.

```
## coverage rate for 95% confidence interval
mean(ci95[, 1] <= 1 & ci95[, 2] >= 1)

## [1] 0.9482

## coverage rate for 90% confidence interval
mean(ci90[, 1] <= 1 & ci90[, 2] >= 1)

## [1] 0.9038
```

As another illustration, we use the polling example described earlier. Again, over repeated random sampling, 95% of the 95% confidence intervals should contain the true parameter value. As the sample size increases, we should observe that the approximation improves with the coverage probability approaching its nominal rate. In the code chunk below, we use a double loop. The outer loop is defined for different sample sizes and the inner loop conducts a simulation and examines, for each simulation, whether the confidence interval contains the truth.

```
p <- 0.6 # true parameter value
n <- c(10, 100, 1000) # 3 sample sizes to be examined
alpha <- 0.05
sims <- 5000 # number of simulations
results <- rep(NA, length(n)) # a container for results
## loop for different sample sizes
for (i in 1:length(n)) {
    ci.results <- rep(NA, sims) # a container for whether CI includes truth
    ## loop for repeated hypothetical survey sampling
    for (j in 1:sims) {
        data <- rbinom(n[i], size = 1, prob = p) # simple  random sampling
        x.bar <- mean(data) # sample proportion as  an estimate
        s.e. <- sqrt(x.bar * (1 - x.bar) / n[i]) # standard errors
        ci.lower <- x.bar - qnorm(1 - alpha / 2) * s.e.
        ci.upper <- x.bar + qnorm(1 - alpha / 2) * s.e.
        ci.results[j] <- (p >= ci.lower) & (p <= ci.upper)
    }
    ## proportion of CIs that contain the true value
    results[i] <- mean(ci.results)
}
```

```
est.regular <- mean(STAR$g4reading[STAR$classtype == 2], na.rm = TRUE)
se.regular <- sd(STAR$g4reading[STAR$classtype == 2], na.rm = TRUE) /
    sqrt(n.regular)
est.regular
## [1] 719.89
se.regular
## [1] 1.83885
```

How should one construct a confidence interval for each estimate? As before, we can rely on the central limit theorem and obtain an approximate confidence interval for each estimate.

```
alpha <- 0.05
## 95% confidence intervals for small class
ci.small <- c(est.small - qnorm(1 - alpha / 2) * se.small,
              est.small + qnorm(1 - alpha / 2) * se.small)
ci.small
## [1] 719.6417 727.1406
## 95% confidence intervals for regular class
ci.regular <- c(est.regular - qnorm(1 - alpha / 2) * se.regular,
                est.regular + qnorm(1 - alpha / 2) * se.regular)
ci.regular
## [1] 716.2859 723.4940
```

These confidence intervals overlap with each other. Does this mean that the estimated average difference between the two groups, or the estimated PATE of small class size, is not statistically significant? An estimated effect is statistically significant if it reflects true patterns in the population, rather than arising from mere chance. To find out the answer to this question, it would be best to compute the confidence interval directly for the estimated average difference. Recall the standard error of the difference-in-means estimator given in equation (7.10). Using this standard error formula, we can compute the 95% confidence interval for the estimated PATE.

```
## difference-in-means estimator
ate.est <- est.small - est.regular
ate.est
## [1] 3.501232
## standard error and 95% confidence interval
ate.se <- sqrt(se.small^2 + se.regular^2)
ate.se
## [1] 2.653485
ate.ci <- c(ate.est - qnorm(1 - alpha / 2) * ate.se,
            ate.est + qnorm(1 - alpha / 2) * ate.se)
ate.ci
## [1] -1.699503  8.701968
```

We find that the average treatment effect of small class size on the fourth-grade reading score is estimated to be 3.50 with a standard error of 2.65. The 95% confidence interval is $[-1.70, 8.70]$, containing zero. This finding suggests that although the estimated average treatment effect is positive, it features a considerable degree of uncertainty.

7.1.6 ANALYSIS BASED ON STUDENT'S t-DISTRIBUTION

The calculation of confidence intervals has so far relied upon the central limit theorem. This is why we used the quantiles of the standard normal distribution when computing confidence intervals, assuming that we have a large enough sample to invoke the central limit theorem. This assumption is useful because the central limit theorem applies to a wide variety of distributions. Given that we often do not know the distribution of an outcome variable, the procedure of constructing confidence intervals described earlier is quite general.

Here, we consider an alternative assumption, that the outcome variable (rather than its sample mean) is generated from a normal distribution. As an illustration, we apply this assumption to the STAR experiment just analyzed in section 7.1.5. We assume that the test scores for each group follow a normal distribution, with possibly different means and variances. While the histograms shown earlier suggest that the distribution of test scores for each group may not satisfy this assumption, the inference resulting from this assumption proves more conservative than the asymptotic inference we have been using based on the central limit theorem. Because many researchers prefer conservative inferences, they often use confidence intervals under this normally distributed outcome assumption even when the assumption is not justifiable.

When a random variable is normally distributed, we can obtain an exact confidence interval for the sample mean using *Student's t-distribution*, also simply called the *t-distribution*. The name of the distribution originates from the fact that its British creator William Gossett, a researcher at beer producer Guinness, published the paper introducing it under the pseudonym "Student." We use t_ν to represent the t-distribution with ν degrees of freedom. Specifically, the z-score of the sample mean is called the *t-statistic* and is distributed according to the t-distribution with $n-1$ *degrees of freedom*. Roughly, the degrees of freedom represent the number of independent observations used for estimation minus the number of parameters to be estimated (see section 4.3.2). The current case involves one parameter to estimate: we use the standard error to estimate the standard deviation of the sampling distribution. This result holds exactly so we do not resort to asymptotic approximation.

```
t.ci <- t.test(STAR$g4reading[STAR$classtype == 1],
               STAR$g4reading[STAR$classtype == 2])
t.ci
##
##  Welch Two Sample t-test
##
## data:STAR$g4reading[STAR$classtype==1] and STAR$g4reading[STAR$classtype == 2]
## t = 1.3195, df = 1541.2, p-value = 0.1872
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.703591  8.706055
## sample estimates:
## mean of x mean of y
##  723.3912  719.8900
```

The degrees of freedom are calculated as 1541.2. Because the size of our sample is not too small, the resulting confidence interval is only slightly wider than the one based on the normal approximation reported above.

7.2 Hypothesis Testing

In section 6.1.5, we presented an analysis of Arnold Schwarzenegger's 2009 veto message to the California legislature, and showed that the particular order of words in his message was highly unlikely to be a consequence of coincidence alone. This was done by examining the likelihood of observing the event that actually happened under a particular probability model. In section 6.6.3, a similar method was used to detect election fraud in Russia, where we generated hypothetical election results and compared them with the actual election outcome to investigate whether the latter was anomalous. In this section, we formalize this logic and introduce a general principle of statistical *hypothesis testing* that underlies such analysis. This principle enables us to determine whether or not the occurrence of an observed event is likely to be due to chance alone.

7.2.1 TEA-TASTING EXPERIMENT

In his classic book *The Design of Experiments*, Ronald Fisher introduced the idea of a statistical hypothesis test. During an afternoon tea party at the University of Cambridge, a lady declared that tea tastes different depending on whether the tea is poured into the milk or the milk is poured into the tea. Fisher examined this claim by using a randomized experiment in which 8 identical cups were prepared and 4 were randomly selected for milk to be poured into the tea. For the remaining 4 cups, the milk was poured first. The lady was then asked to identify, for each cup, whether the tea or the milk had been poured first. To everyone's surprise, the lady correctly classified all the cups. Did this happen by luck or did the lady actually possess the ability to detect the order, as she claimed?

Table 7.2. Tea-Tasting Experiment.

Cups	*Lady's guess*	*Actual order*	*Scenarios*			. . .
1	M	M	T	T	T	
2	T	T	T	T	M	
3	T	T	T	T	M	
4	M	M	T	M	M	
5	M	M	M	M	T	
6	T	T	M	M	T	
7	T	T	M	T	M	
8	M	M	M	M	T	
Number of correct guesses		8	4	6	2	. . .

Note: "M" and "T" represent two scenarios, "milk is poured first" and "tea is poured first," respectively. Under the hypothesis that the lady has no ability to distinguish the order in which milk and tea were poured into each cup, her guess will be identical regardless of which cups had milk/tea poured first.

To analyze this randomized experiment, we draw on potential outcomes as explained in chapter 2. For each of the 8 cups, we consider two potential guesses given by the lady, which may or may not depend on whether milk or tea was actually poured into the cup first. If we hypothesize that the lady had no ability to distinguish whether milk or tea was poured into the cup first, then her guess should not depend on the actual order in which milk and tea were poured. In other words, under this hypothesis, the two potential outcomes should be identical. Recall the *fundamental problem of causal inference*, which states that only one of the two potential outcomes can be observed. Here, the hypothesis that the lady possesses no ability to distinguish the two types of tea with milk reveals her responses under counterfactual scenarios.

Fisher's analysis proceeds under this hypothesis and involves computing the number of correctly guessed cups under every possible assignment combination. As discussed in section 7.1.1, this experiment is an example of *complete randomization*, where the number of observations assigned to each condition is fixed a priori. In contrast, *simple randomization* would randomize each cup independently without such a constraint. Table 7.2 illustrates Fisher's method. The second column of the table shows the lady's actual guess for each cup, which is identical to the true order (third column) in which milk and tea were poured into the cup. In the remaining columns, we show three arbitrarily selected combinations of assigning 4 cups to "milk first" and the other 4 to "tea first." Although these counterfactual assignment combinations did not occur in the actual experiment, we can compute the number of correctly guessed cups under each scenario with the aforementioned hypothesis that the lady lacks the ability to distinguish between the two types of tea with milk and thus different assignments do not affect the lady's guess. This is done by simply comparing the lady's guess (second column), which is assumed to remain unchanged, with each counterfactual assignment. For example, if the cups had received the assignments in the fifth column of the table, then the number of correctly classified cups would have been 6.

```
## comparison with analytical answers; the differences are small
prop.table(table(correct)) - true

## correct
##             0             2             4             6
##  0.0007142857 -0.0015714286 -0.0142857143  0.0194285714
##             8
## -0.0042857143
```

The major advantage of Fisher's analysis is that the inference is solely based on the randomization of treatment assignment. Such inference is called *randomization inference*. Methods based on randomization inference typically do not require a strong assumption about the data-generating process because researchers control the randomization of treatment assignment, which alone serves as the basis of inference.

7.2.2 THE GENERAL FRAMEWORK

The tea-tasting experiment described above illustrates a general framework called statistical hypothesis testing. Statistical hypothesis testing is based on probabilistic *proof by contradiction*. Proof by contradiction is a general strategy of mathematical proof in which one demonstrates that assuming the contrary of what we would like to prove leads to a logical contradiction. For example, consider the proposition that there is no smallest positive *rational number*. To prove this proposition, we assume that the conclusion is false. That is, suppose that there exists a smallest positive rational number a. Recall that any rational number can be expressed as the fraction of two integers: $a = p/q > 0$ where both the numerator p and the nonzero denominator q are positive integers. But, for example, $b = a/2$ is smaller than a, and yet b is also a rational number. This contradicts the hypothesis that a is the smallest positive rational number.

In the case of statistical hypothesis testing, we can never reject a hypothesis with 100% certainty. Consequently, we use a probabilistic version of proof by contradiction. We begin by assuming a hypothesis we would like to eventually refute. This hypothesis is called a *null hypothesis*, often denoted by H_0. In the current application, the null hypothesis is that the lady has no ability to tell whether milk or tea is poured first into a cup. This is an example of *sharp null hypothesis* because all potential outcomes for each observation are determined, and therefore known, under this hypothesis. In contrast, we will later consider a nonsharp null hypothesis, which fixes the *average* potential outcome rather than every potential outcome.

Second, we choose a *test statistic*, which is some function of observed data. For the tea-tasting experiment, the test statistic is the number of correctly specified cups. Next, under the null hypothesis, we derive the *sampling distribution* of the test statistic, which is given in figure 7.2 for our application. This distribution is also called the *reference distribution*. Finally, we ask whether the observed value of the test statistic

Table 7.3. Type I and Type II Errors in Hypothesis Testing.

	Reject H_0	Retain H_0
H_0 is true	**type I error**	correct
H_0 is false	correct	**type II error**

Note: H_0 represents the null hypothesis.

is likely to occur under the reference distribution. In the current experiment, the number of correctly classified cups is observed to be 8. If 8 is likely under the reference distribution, we retain the null hypothesis. If it is unlikely, then we reject the null hypothesis.

In this textbook, we prefer to use phrases such as "fail to reject the null hypothesis" and "retain the null hypothesis" instead of "accept the null hypothesis." Philosophical views on this issue differ, but we adopt a perspective that failure to reject the null hypothesis is evidence for some degree of consistency between the data and the hypothesis, but does not necessarily indicate the correctness of the null hypothesis. Others, however, argue that the failure to reject the null hypothesis implies acceptance of the hypothesis. Regardless of one's stance on this issue, statistical hypothesis testing provides empirical support for scientific theories.

How should we quantify the degree to which the observed value of the test statistic is unlikely to occur under the null hypothesis? We use the *p-value* for this purpose. The p-value can be understood as the probability that under the null hypothesis, we observe a value of the test statistic at least as extreme as the one we actually observed. A smaller p-value provides stronger evidence against the null hypothesis. Importantly, the p-value does not represent the probability that the null hypothesis is true. This probability is actually either 1 or 0 because the null hypothesis is either true or false, though researchers do not know which.

In order to decide whether or not to reject the null hypothesis, we must specify the *level of test* α (as explained later, this α is the same as the confidence level α for confidence intervals discussed earlier). If the p-value is less than or equal to α, then we reject the null hypothesis. The level of test represents the probability of false rejection if the null hypothesis is true. This error is called *type I error*. Typically, we would like the level of test to be low. Commonly used values of α are 0.05 and 0.01.

Table 7.3 shows two types of errors in hypothesis testing. While researchers can specify the degree of type I error by choosing the level of test α, it is not possible to directly control *type II error*, which results when researchers retain a false null hypothesis. Notably, there is a clear trade-off between type I and type II errors in that minimizing type I error usually increases the risk of type II error. As an extreme example, suppose that we never reject the null hypothesis. Under this scenario, the probability of type I error is 0 if the null hypothesis is true, but the probability of type II error is 1 if the null hypothesis is false.

In the case of the tea-tasting experiment, the test statistic is the number of correctly classified cups. Since the observed value of this test statistic was 8, which is the most extreme value, the p-value equals the probability that the number of correct guesses is 8 or $1/70 \approx 0.014$. *If the lady correctly classified 6 cups instead of 8, two values are at* double one of the areas to obtain the two-sided p-value. Note that this may not work in other cases where the reference distribution is not symmetric.

```
2 * upper
## [1] 0.01016866
```

If, on the other hand, our alternative hypothesis is $p > 0.5$ rather than $p \neq 0.5$, then we must compute the one-sided p-value. In this case, there is no need to consider the possibility of an extremely small value because the alternative hypothesis specifies p to be greater than the null value. Hence, the one-sided p-value is given by the blue area under the curve above the observed value in the figure.

```
## one-sided p-value
upper
## [1] 0.005084332
```

Regardless of whether we use the one-sided or two-sided p-value, we reject the null hypothesis that Obama's support in the population is exactly 50%. We conclude that the 4 percentage point difference we observe is unlikely to arise due to chance alone.

When using the normal distribution as the reference distribution, researchers often use the z-score to standardize the test statistic by subtracting its mean and dividing it by its standard deviation. Once this transformation is made, the reference distribution becomes the standard normal distribution. That is, if we use μ_0 to denote the hypothesized mean under the null hypothesis, we have the following result so long as the sample size is sufficiently large (due to the central limit theorem):

$$\frac{\overline{X}_n - \mu_0}{\text{standard error of } \overline{X}_n} \sim \mathcal{N}(0, 1). \tag{7.19}$$

Note that this transformation does not change the outcome of the hypothesis testing conducted above. In fact, the p-value will be identical with or without this transformation. However, one can easily compare the z-score with the critical values shown in table 7.1 in order to determine whether to reject the null hypothesis without computing the p-value. For example, under the two-sided alternative hypothesis, if the z-score is greater than 1.96, then we reject the null hypothesis. We now show, using the current example, that we obtain the same p-value as above.

```
z.score <- (x.bar - 0.5) / se
z.score
## [1] 2.57004
pnorm(z.score, lower.tail = FALSE) # one-sided p-value
## [1] 0.005084332
```

```
2 * pnorm(z.score, lower.tail = FALSE) # two-sided p-value
## [1] 0.01016866
```

This test, which is based on the z-score of the sample mean, is called the *one-sample z-test*. Although we used this test for a Bernoulli random variable in this example, the test can be applied to a wide range of nonbinary random variables so long as the sample size is sufficiently large and the central limit theorem is applicable. For nonbinary random variables, we will use the sample variance to estimate the standard error. If the random variable X is distributed according to the normal distribution, then the same test statistic, i.e., the z-score of the sample mean, follows the t-distribution with $n-1$ degrees of freedom instead of the standard normal distribution. This *one-sample t-test* is more conservative than the one-sample z-test, meaning that the former gives a greater p-value than the latter. Some researchers prefer conservative inference and hence use the one-sample t-test rather than the one-sample z-test.

Suppose that $\{X_1, X_2, \ldots, X_n\}$ are n independently and identically distributed random variables with mean μ and variance σ^2. The **one-sample z-test** consists of the following components:

1. Null hypothesis that the population mean μ is equal to a prespecified value μ_0: $H_0 : \mu = \mu_0$
2. Alternative hypothesis: $H_1 : \mu \neq \mu_0$ (two-sided), $H_1 : \mu > \mu_0$ (one-sided), or $H_1 : \mu < \mu_0$ (one-sided)
3. Test statistic (z-statistic): $Z_n = (\overline{X}_n - \mu_0)/\sqrt{\hat{\sigma}^2/n}$, where $\overline{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$ (sample mean)
4. Reference distribution: $Z_n \sim \mathcal{N}(0, 1)$ when n is large
5. Variance: $\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \overline{X}_n)^2$ (sample variance) or $\hat{\sigma}^2 = \mu_0(1 - \mu_0)$ if X is a Bernoulli random variable
6. p-value: $\Phi(-|Z_n|)$ (one-sided) and $2\Phi(-|Z_n|)$ (two-sided), where $\Phi(\cdot)$ is the cumulative distribution function (CDF) of the standard normal distribution

If X is normally distributed, the same test statistic Z_n is called the t-statistic and follows the t-distribution with $n-1$ degrees of freedom. The p-value will be based on the cumulative distribution of this t-distribution. This is called the **one-sample t-test**, which is more conservative than the one-sample z-test.

There exists a general one-to-one relationship between confidence intervals and hypothesis tests. Compare equation (7.19) with equation (7.15). The difference is that the unknown population mean $\mathbb{E}(X)$ in the former is replaced with the hypothesized population mean μ_0 in the latter. Note that under a null hypothesis the hypothesized mean μ_0 represents the actual population mean. This suggests that we reject a null hypothesis $H_0 : \mu = \mu_0$ using the α-level two-sided test if and only if the $(1-\alpha)\times 100\%$

confidence interval does not contain μ_0. We can confirm this result using the current example by checking that 0.5 is contained in the 99% confidence interval (we fail to reject the null hypothesis when $\alpha = 0.01$) but not in the 95% confidence interval (we reject the null when $\alpha = 0.05$).

```
## 99% confidence interval contains 0.5
c(x.bar - qnorm(0.995) * se, x.bar + qnorm(0.995) * se)

## [1] 0.4999093 0.5806408

## 95% confidence interval does not contain 0.5
c(x.bar - qnorm(0.975) * se, x.bar + qnorm(0.975) * se)

## [1] 0.5095605 0.5709896
```

It turns out that this one-to-one relationship between confidence intervals and hypothesis testing holds in general. Many researchers, however, prefer to report confidence intervals rather than p-values because the former also contain information about the magnitude of effects, quantifying *scientific significance* as well as *statistical significance*.

We conducted the one-sample z-test for sample proportion "by hand" above in order to illustrate the underlying idea. However, R has the `prop.test()` function, which enables us to conduct this test in a single line of R code. For the one-sample test of sample proportion like the one above, the function takes the number of successes as the main argument `x` and the number of trials as the argument `n`. In addition, one can specify the success probability under the null hypothesis as `p`, as well as the alternative hypothesis (`"two.sided"` for the two-sided alternative hypothesis, and either `"less"` or `"greater"` for the one-sided alternative hypothesis). The default confidence level is 95%, which we can change with the `conf.level` argument.

Finally, the `correct` argument determines whether a continuity correction should be applied in order to improve the approximation (the default is `TRUE`). This correction is generally recommended, especially when the sample size is small because the binomial distribution, which is a discrete distribution, is approximated by a continuous distribution, i.e., the normal distribution. We first show that `prop.test()` without a continuity correction gives a result identical to the one obtained earlier. We then show the result based on the continuity correction.

```
## no continuity correction to get the same p-value as above
prop.test(550, n = n, p = 0.5, correct = FALSE)

##
##  1-sample proportions test without continuity
##  correction
##
## data:  550 out of n, null probability 0.5
## X-squared = 6.6051, df = 1, p-value = 0.01017
## alternative hypothesis: true p is not equal to 0.5
## 95 percent confidence interval:
##  0.5095661 0.5706812
## sample estimates:
##        p
## 0.540275

## with continuity correction
prop.test(550, n = n, p = 0.5)

##
##  1-sample proportions test with continuity correction
##
## data:  550 out of n, null probability 0.5
## X-squared = 6.445, df = 1, p-value = 0.01113
## alternative hypothesis: true p is not equal to 0.5
## 95 percent confidence interval:
##  0.5090744 0.5711680
## sample estimates:
##        p
## 0.540275
```

The `prop.test()` function also conveniently yields confidence intervals. Note that the standard error used for confidence intervals is different from the standard error used for hypothesis testing. This is because the latter standard error is derived under the null hypothesis $\sqrt{p(1-p)/n}$, whereas the standard error for confidence intervals is computed using the estimated proportion, $\sqrt{\overline{X}_n(1-\overline{X}_n)/n}$. To illustrate a different level of confidence intervals, we can compute 99% confidence intervals using the `conf.level` argument.

```
prop.test(550, n = n, p = 0.5, conf.level = 0.99)

##
##  1-sample proportions test with continuity correction
##
## data:  550 out of n, null probability 0.5
## X-squared = 6.445, df = 1, p-value = 0.01113
## alternative hypothesis: true p is not equal to 0.5
## 99 percent confidence interval:
##  0.4994182 0.5806040
## sample estimates:
##        p
## 0.540275
```

As another example, we revisit the analysis of the STAR project given in section 7.1.5. We first conduct a one-sample t-test just for illustration. Suppose that we test the null hypothesis that the population mean test score is 710, i.e., $H_0 : \mu = 710$.

We use the `t.test()` function where we specify the null value μ_0 using the `mu` argument. The other arguments such as `alternative` and `conf.level` work in the exact same way as for the `prop.test()` function. We use the reading test score for our analysis and conduct a two-sided one-sample t-test. As the result below shows, we retain, at the 0.05 level, the null hypothesis that the population mean of test score is 710. The resulting p-value is small, leading to the rejection of the null hypothesis.

```
## two-sided one-sample t-test
t.test(STAR$g4reading, mu = 710)

##
##  One Sample t-test
##
## data:  STAR$g4reading
## t = 10.407, df = 2352, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 710
## 95 percent confidence interval:
##  719.1284 723.3671
## sample estimates:
## mean of x
##  721.2478
```

7.2.4 TWO-SAMPLE TESTS

We now move to a more realistic analysis of the STAR project. When analyzing randomized controlled trials like this, researchers often conduct a statistical hypothesis test with the null hypothesis that the population average treatment effect (PATE) is zero, i.e., $H_0 : \mathbb{E}(Y_i(1) - Y_i(0)) = 0$ with a two-sided alternative hypothesis given by $H_1 : \mathbb{E}(Y_i(1) - Y_i(0)) \neq 0$. If we assume that the PATE cannot be negative, then we employ a one-sided alternative hypothesis, $H_1 : \mathbb{E}(Y_i(1) - Y_i(0)) > 0$. In contrast, if we assume that the PATE cannot be positive, we set $H_1 : \mathbb{E}(Y_i(1) - Y_i(0)) < 0$. In this application, we would like to test whether or not the PATE of small class size on the grade-four reading score (relative to regular class size) is zero.

To test this null hypothesis, we use the difference-in-means estimator as a test statistic. More generally, beyond randomized controlled trials, we can use the two-sample tests based on the difference-in-means estimator to investigate the null hypothesis that the means are equal between these two populations. What is the reference distribution of this test statistic? We can approximate it by appealing to the central limit theorem as in section 7.1.5. The theorem implies that the sample means of the treatment and control groups have a normal distribution. Therefore, under the null hypothesis of equal means between the two populations, the difference between these two sample means is also normally distributed with mean zero. Furthermore, the z-score of the difference in sample means follows the standard normal distribution. We can use this fact to conduct the *two-sample z-test* (see equation (7.18) for the expression of standard error, which serves as the denominator of the test statistic). As in the one-sample tests, if the outcomes are assumed to be normally distributed, the *two-sample t-test* can be used, which yields a more conservative inference.

Suppose that $\{X_1, X_2, \ldots, X_{n_0}\}$ represent n_0 independently and identically distributed random variables with mean μ_0 and variance σ_0^2. Similarly, $\{Y_1, Y_2, \ldots, Y_{n_1}\}$ represent n_1 independently and identically distributed random variables with mean μ_1 and variance σ_1^2. The **two-sample** z**-test** of sample means consists of the following components:

1. Null hypothesis that two populations have the same mean: $H_0 : \mu_0 = \mu_1$
2. Alternative hypothesis: $H_1 : \mu_0 \neq \mu_1$ (two-sided), $H_1 : \mu_0 > \mu_1$ (one-sided), or $H_1 : \mu_0 < \mu_1$ (one-sided)
3. Test statistic (z-statistic): $Z_n = (\overline{Y}_{n_1} - \overline{X}_{n_0}) / \sqrt{\frac{1}{n_1}\hat{\sigma}_1^2 + \frac{1}{n_0}\hat{\sigma}_0^2}$
4. Reference distribution: $Z_n \sim \mathcal{N}(0, 1)$ when n_0 and n_1 are large
5. Variance: $\hat{\sigma}_0^2 = \frac{1}{n_0-1}\sum_{i=1}^{n_0}(X_i - \overline{X}_{n_0})^2$ and $\hat{\sigma}_1^2 = \frac{1}{n_1-1}\sum_{i=1}^{n_1}(Y_i - \overline{Y}_{n_1})^2$ (sample variances) or $\hat{\sigma}_0^2 = \hat{\sigma}_1^2 = \hat{p}(1-\hat{p})$ with $\hat{p} = \frac{n_0}{n_0+n_1}\overline{X}_{n_0} + \frac{n_1}{n_0+n_1}\overline{Y}_{n_1}$ if X and Y are Bernoulli random variables
6. p-value: $\Phi(-|Z_n|)$ (one-sided) and $2\Phi(-|Z_n|)$ (two-sided), where $\Phi(\cdot)$ is the cumulative distribution function (CDF) of the standard normal distribution

If X and Y are normally distributed, the same test statistic Z_n is called the t-statistic and follows the t-distribution. The p-value will be based on the cumulative distribution of this t-distribution. This is called the **two-sample** t**-test**, which is more conservative than the one-sample z-test.

Recall from section 7.1.5 that the estimated PATE is stored as an R object `ate.est` whereas its standard error is given by the R object `ate.se`. Using these objects, we compute the one-sided and two-sided p-values as follows.

```
## one-sided p-value
pnorm(-abs(ate.est), mean = 0, sd = ate.se)

## [1] 0.09350361

## two-sided p-value
2 * pnorm(-abs(ate.est), mean = 0, sd = ate.se)

## [1] 0.1870072
```

Since this p-value is much greater than the typical threshold of 5%, we cannot reject the hypothesis that the average treatment effect of small class size on the fourth-grade reading test score is zero.

The hypothesis test conducted above is based on the large sample approximation because we relied upon the central limit theorem to derive the reference distribution. Similar to the discussion in section 7.1.5, if we assume that the outcome variable is normally distributed, then we could use the t-distribution instead of the normal distribution to conduct a hypothesis test. As a test statistic, we use the z-score for the difference-in-means estimator, which is called the t-statistic in the case of this two-sample t-test. Unlike the one-sample example discussed in section 7.1.5, however, the degrees of freedom must be approximated for the *two-sample t-test*. Because the t-distribution generally has heavier tails than the normal distribution, the *t-test* is more conservative and hence is often preferred even when the outcome variable may not be normally distributed.

In R, we can conduct a two-sample t-test using the `t.test()` function as we did for a one-sample t-test. For the two-sample t-test, the function takes two vectors, each of which contains data for one of the two groups. We can specify the difference between the means of the two groups, or the PATE in this application, under the null hypothesis via the `mu` argument. The default value for this argument is zero, which is what we would like to use in the current example.

```
## testing the null of zero average treatment effect
t.test(STAR$g4reading[STAR$classtype == 1],
       STAR$g4reading[STAR$classtype == 2])
##
##  Welch Two Sample t-test
##
## data:STAR$g4reading[STAR$classtype==1] and STAR$g4reading[STAR$classtype == 2]
## t = 1.3195, df = 1541.2, p-value = 0.1872
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.703591  8.706055
## sample estimates:
## mean of x mean of y
##  723.3912  719.8900
```

The output displays the value of the t-statistic as well as the p-value and the degrees of freedom for Student's t-distribution used for the test. Since the p-value is greater than the standard threshold of $\alpha = 0.05$, we fail to reject the null hypothesis that the average treatment effect of small class size on the fourth-grade reading score is zero. As in the case of `prop.test()`, the output of the `t.test()` function contains the confidence interval for the corresponding level. As expected from the use of the t-distribution, this confidence interval is slightly wider than the confidence interval based on the normal approximation we obtained in section 7.1.5. The confidence interval also contains zero, which is consistent with the fact that we fail to reject the null hypothesis of zero average treatment effect.

As another application of hypothesis tests, we reanalyze the labor market discrimination experiment described in section 2.1. In this experiment, fictitious résumés of job applicants were sent to potential employers. Researchers randomly assigned stereotypically African-American or Caucasian names to each résumé and examined whether or not the callback rate depended on the race of the applicant. The data set we analyze is contained in the CSV file `resume.csv`. The names and descriptions of variables in this data set are given in table 2.1. The outcome variable of interest is `call`, which indicates whether or not each résumé received a callback. The treatment variable is the race of the applicant, `race`, and we focus on the comparison between black-sounding and white-sounding names.

We test the null hypothesis that the probability of receiving a callback is the same between résumés with black-sounding names and those with white-sounding names. We use the `prop.test()` function to implement the two-sample z-test. The input is a table whose columns represent the counts of successes and failures and rows represent the two groups to be compared. We will use a one-sided test because résumés with black-sounding names are hypothesized to receive fewer callbacks.

```
resume <- read.csv("resume.csv")
## organize the data in tables
x <- table(resume$race, resume$call)
x

##
##            0    1
##   black 2278  157
##   white 2200  235

## one-sided test
prop.test(x, alternative = "greater")

##
##  2-sample test for equality of proportions with
##  continuity correction
##
## data:  x
## X-squared = 16.449, df = 1, p-value = 2.499e-05
## alternative hypothesis: greater
## 95 percent confidence interval:
##  0.01881967 1.00000000
## sample estimates:
##    prop 1    prop 2
## 0.9355236 0.9034908
```

Thus, the result supports the alternative hypothesis that résumés with white-sounding names are more likely to receive callbacks than those with black-sounding names. It is instructive to directly compute this p-value without using the `prop.test()` function. Under the null hypothesis of equal proportions between the two groups, i.e., $H_0 : \mu_0 = \mu_1$, the standard error of the difference-in-means (or more accurately difference-in-proportions) estimator can be computed as

$$\sqrt{\frac{\widehat{\mathbb{V}(X)}}{n_0} + \frac{\widehat{\mathbb{V}(Y)}}{n_1}} = \sqrt{\frac{\hat{p}(1-\hat{p})}{n_0} + \frac{\hat{p}(1-\hat{p})}{n_1}} = \sqrt{\hat{p}(1-\hat{p})\left(\frac{1}{n_0} + \frac{1}{n_1}\right)}, \quad (7.20)$$

where X and Y are the outcome variables for the résumés with black-sounding and white-sounding names, respectively, n_0 and n_1 are sample sizes, and $\hat{p} = \frac{1}{n_0+n_1}(\sum_{i=1}^{n_0} X_i + \sum_{i=1}^{n_1} Y_i)$ is the overall sample proportion. We use the same estimate $\hat{p}(1 - \hat{p})$ for the variances of X and Y because under the null hypothesis of identical proportions, their variances, which are based on the proportions, are also identical.

```
## sample size
n0 <- sum(resume$race == "black")
n1 <- sum(resume$race == "white")
## sample proportions
p <- mean(resume$call) # overall
p0 <- mean(resume$call[resume$race == "black"]) # black
p1 <- mean(resume$call[resume$race == "white"]) # white
## point estimate
est <- p1 - p0
est

## [1] 0.03203285

## standard error
se <- sqrt(p * (1 - p) * (1 / n0 + 1 / n1))
se

## [1] 0.007796894

## z-statistic
zstat <- est / se
zstat

## [1] 4.108412

## one-sided p-value
pnorm(-abs(zstat))

## [1] 1.991943e-05
```

The exact same p-value can be obtained using the `prop.test()` function without a continuity correction.

```
prop.test(x, alternative = "greater", correct = FALSE)
##
##  2-sample test for equality of proportions without
##  continuity correction
##
## data:  x
## X-squared = 16.879, df = 1, p-value = 1.992e-05
## alternative hypothesis: greater
## 95 percent confidence interval:
```

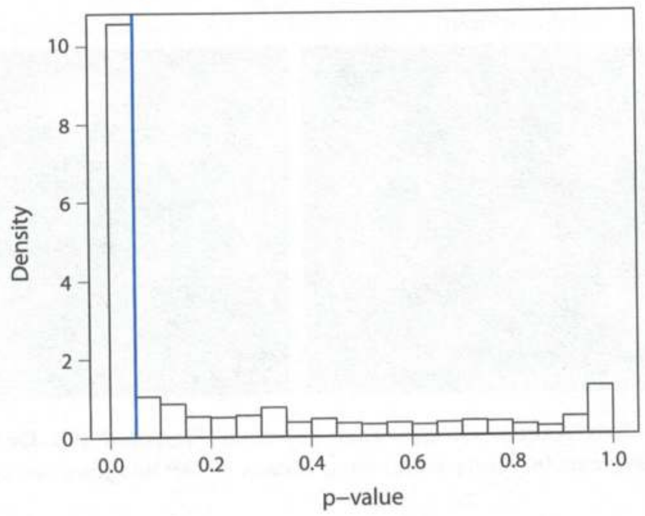


Figure 7.4. The Distribution of p-Values for Hypothesis Tests Published in Two Leading Political Science Journals.

```
##  0.01923035 1.00000000
## sample estimates:
##    prop 1    prop 2
## 0.9355236 0.9034908
```

7.2.5 PITFALLS OF HYPOTHESIS TESTING

Since Fisher's tea-tasting experiment, hypothesis testing has been extensively used in the scientific community to determine whether or not empirical findings are statistically significant. Statistical hypothesis testing represents a rigorous methodology to draw a conclusion in the presence of uncertainty. However, the prevalent use of hypothesis testing also leads to *publication bias* because only statistically significant results, and especially the ones that are surprising to the scientific community, tend to be published. In many social science journals, the α-level of 5% is regarded as the cutoff that determines whether empirical findings are statistically significant or not. As a result, researchers tend to submit their papers to journals only when their empirical results have p-values smaller than this 5% threshold. In addition, journals may also be more likely to publish statistically significant results than nonsignificant results. This is problematic because even if the null hypothesis is true, researchers have a 5% chance of obtaining a p-value less than 5%.

In one study, two researchers examined more than 100 articles published in the two leading political science journals over a decade or so.[2] The researchers collected the p-values for the hypotheses tested in those articles. Figure 7.4 shows that a majority of reported findings have p-values less than or equal to the 5% threshold, which is

[2] Alan Gerber and Neil Malhotra (2008) "Do statistical reporting standards affect what is published? Publication bias in two leading political science journals." *Quarterly Journal of Political Science*, vol. 3, no. 3, pp. 313–326.

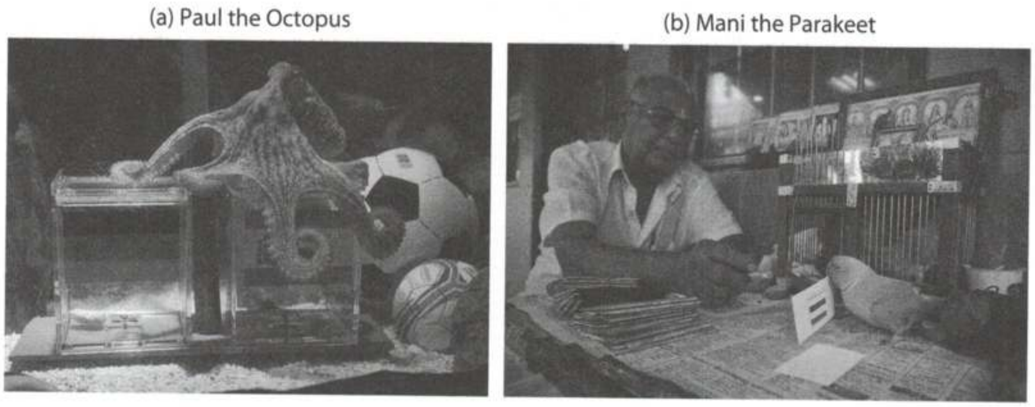


Figure 7.5. Two Animal Oracles that Correctly Predicted the Outcomes of Soccer Matches. Sources: (a) Reuters/Wolfgang Rattay. (b) AP Images/Joan Leong.

indicated by the blue vertical line. In addition, there appears to be a discontinuous jump at the threshold, suggesting that journals are publishing more empirical results that are just below the threshold than results just above it.

Another important pitfall regarding hypothesis testing is *multiple testing*. Recall that statistical hypothesis testing is probabilistic. We never know with 100% certainty whether the null hypothesis is true. Instead, as explained earlier, we typically have type I and type II errors when conducting hypothesis tests (see table 7.3). Multiple testing problems refer to the possibility of *false discoveries* when testing multiple hypotheses.

To see this, suppose that a researcher tests 10 hypotheses when, unbeknown to the researcher, all of these hypotheses are in fact false. What is the probability that the researcher rejects at least one null hypothesis using 5% as the threshold? If we assume independence among these hypotheses tests, we can compute this probability as

$$\begin{aligned} P(\text{reject at least one hypothesis}) &= 1 - P(\text{reject no hypothesis}) \\ &= 1 - 0.95^{10} \approx 0.40. \end{aligned}$$

The second equality follows because the probability of not rejecting the null hypothesis when the null hypothesis is true is $1 - \alpha = 0.95$ and we assume independence among these 10 hypothesis tests. Thus, the researcher has a 40% chance of making at least one false discovery. The lesson here is that if we conduct many hypothesis tests, we are likely to falsely find statistically significant results.

To illustrate the multiple testing problem, consider "Paul the Octopus" shown in figure 7.5a. This octopus in a German aquarium attracted media attention during the 2010 World Cup soccer tournament by correctly predicting all seven matches involving Germany, as well as the outcome of the final match between the Netherlands and Spain. Paul predicted by choosing to enter one of two containers with a country flag as shown in the figure. Given this data, we can conduct a hypothesis test with the null hypothesis that Paul does not possess any ability to predict soccer matches. Under this null hypothesis, Paul randomly guesses a winner out of two countries in question. What is the probability that Paul correctly predicts the outcomes of all 8 matches? Since Paul has a 50% chance of correctly predicting each match, this one-sided p-value is equal to $1/2^8 \approx 0.004$. This value is well below the usual 5% threshold and hence can be considered statistically significant.

However, the problem of multiple testing suggests that if we have many animals predict soccer matches, we are likely to find an animal that appears to be prophetic. During the same world cup, another animal, "Mani the Parakeet" shown in figure 7.5b, was reported to have a similar oracle ability. The parakeet correctly predicted only 6 out of 8 matches. Each time, he selected one of two pieces of paper with his beak and flipped it to reveal a winner, without viewing country flags as Paul did. Since no scientific theory suggests animals can possess such predictive ability, we may conclude that Paul and Mani represent false discoveries due to the problem of multiple testing. Although beyond the scope of this book, statisticians have developed various methods that make appropriate adjustments for multiple testing.

The **multiple testing problem** is that conducting many hypothesis tests is likely to result in false discoveries, i.e., incorrect rejection of null hypotheses.

7.2.6 POWER ANALYSIS

Another problem of hypothesis testing is that null hypotheses are often not interesting. For example, who would believe that the small class in the STAR study has *exactly* zero average causal effect on students' test scores as assumed under the null hypothesis? The effect size might be small, but it is hard to imagine that it is exactly zero. A related problem is that failure to reject the null hypothesis does not necessarily mean that the null hypothesis is true. Failure to reject the null may arise because data are not informative about the null hypothesis. For example, if the sample size is too small, then even if the true average treatment effect is not zero, researchers may fail to reject the null hypothesis of zero average effect because the standard error is too large.

We use *power analysis* in order to formalize the degree of informativeness of data in hypothesis tests. The *power* of a statistical hypothesis test is defined as one minus the probability of *type II error*:

$$\text{power} = 1 - P(\text{type II error}).$$

Recall from the discussion in section 7.2.2 that type II error occurs when researchers retain a false null hypothesis. Therefore, we would like to maximize the power of a statistical hypothesis test so that we can detect departure from the null hypothesis as much as possible.

Power analysis is often used to determine the smallest sample size necessary to estimate the parameter with enough precision that its observed value is distinguishable from the parameter value assumed under the null hypothesis. This is typically done as part of research planning in order to inform data collection. In sample surveys, for example, researchers wish to know the number of people they must interview in order to reject the null hypothesis of an exact tie in support level when one candidate is ahead of the other by a prespecified degree (see also the discussion in section 7.1.4). Moreover, experimentalists use power analysis to compute the number of observations necessary

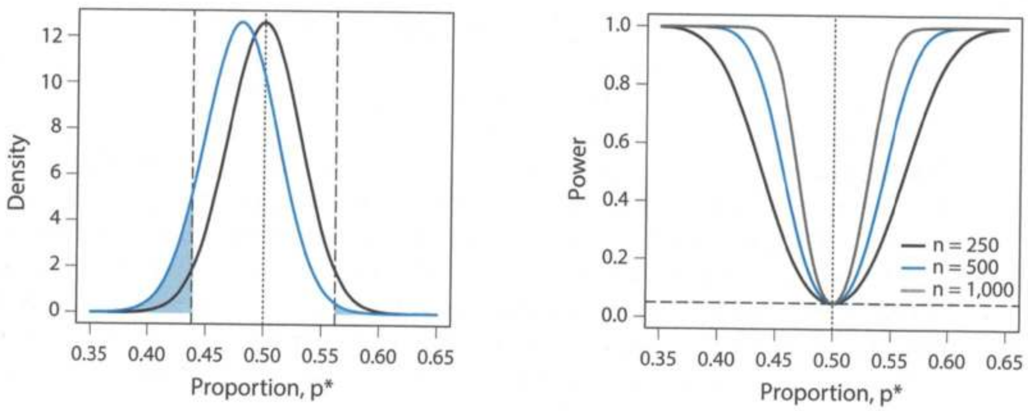


Figure 7.6. Illustration of Power Analysis. In the left-hand plot, the solid black line represents the sampling distribution of sample proportion under the null hypothesis $p = 0.5$ (vertical dotted line). The blue solid line represents the sampling distribution of the test statistic under a hypothetical data-generating process, which has mean 0.48. The sum of the two blue shaded areas equals the power of this statistical test when the significance level is $\alpha = 0.05$. The vertical dashed lines represent thresholds, above or below which the null hypothesis will be rejected. The right-hand plot displays the power function under the same setting with three different sample sizes.

to reject the null hypothesis of zero average treatment effect when the effect is actually not zero. As a result, power analysis is often required for research grant applications in order to justify the budget that researchers are requesting.

Again, we use survey sampling as an example. Suppose that we wish to find out how many respondents we must interview to be able to reject the null hypothesis that the support level for Obama, denoted by p, is exactly 50% when the true support level is at least 2 percentage points away from an exact tie, i.e., 48% or less, or 52% or greater. That is, 2 percentage points is the smallest deviation from the null hypothesis we would like to detect with a high probability. Further assume that we will use the sample proportion as the test statistic, and that the significance level is set to $\alpha = 0.05$ with a two-sided alternative hypothesis.

To compute the power, we need to consider two sampling distributions of the test statistic. The first is the sampling distribution under the null distribution. We have already derived the large sample approximation of this sampling distribution earlier: $\mathcal{N}(p, p(1-p)/n)$, where p is the null value of the population proportion. In our application, $p = 0.5$. The second is the sampling distribution under a hypothetical data-generating process. In the current case, this distribution is approximated by $\mathcal{N}(p^*, p^*(1-p^*)/n)$ via the *central limit theorem*, where p^* is either less than or equal to 0.48 or greater than or equal to 0.52.

The left-hand plot of figure 7.6 graphically illustrates the mechanics of power analysis in this case. In the plot, the two sampling distributions of the sample proportion, one centered around 0.5 under the null hypothesis (black solid line) and the other centered around 0.48 under a hypothetical data-generating process (blue solid line), are shown. We choose 0.48 as the mean value under the hypothetical data-generating process because any distribution with a mean less than this value would result in greater statistical power, which is the probability of correctly rejecting the null, and hence would require a smaller sample size. For the meantime, we set the sample size n to 250.

Under this setting, we compute the power of the statistical test, which is the probability of rejecting the null hypothesis. To do this, we first derive the thresholds that determine the rejection region. As shown in section 7.2.3, the threshold is equal to the null value p_0 plus or minus the product of the standard error and critical value $z_{\alpha/2}$, i.e., $p_0 \pm z_{\alpha/2} \times \text{standard error}$, where in the current setting $p_0 = 0.5$ and $z_{\alpha/2} \approx 1.96$. In the left-hand plot of the figure, these thresholds are denoted by black dashed lines and we reject the null hypothesis if an observed value is more extreme than they are.

We use the probability distribution indicated by the blue solid line in the figure when computing the probability of rejection under the hypothetical data-generating process. That is, the power of the test equals the sum of the two blue shaded areas in the figure, one large area below the lower threshold and the other small area above the upper threshold. Formally, it is given by

$$\text{power} = P(\overline{X}_n < p - z_{\alpha/2} \times \text{standard error}) + P(\overline{X}_n > p + z_{\alpha/2} \times \text{standard error}).$$

In this equation, the sample proportion $\overline{X}_n$ is assumed to be approximately distributed according to $\mathcal{N}(p^*, p^*(1-p^*)/n)$, where in the current application p^* is set to 0.48. We can compute the power of a test in R as follows.

```
## set the parameters
n <- 250
p.star <- 0.48 # data-generating process
p <- 0.5 # null value
alpha <- 0.05
## critical value
cr.value <- qnorm(1 - alpha / 2)
## standard errors under the hypothetical data-generating process
se.star <- sqrt(p.star * (1 - p.star) / n)
## standard error under the null
se <- sqrt(p * (1 - p) / n)
## power
pnorm(p - cr.value * se, mean = p.star, sd = se.star) +
    pnorm(p + cr.value * se, mean = p.star, sd = se.star, lower.tail = FALSE)

## [1] 0.09673114
```

Under these conditions, the power of the test is only 10%. We can examine how the power of this test changes as a function of the sample size and hypothetical data-generating process. The right-hand plot of figure 7.6 presents the *power function*, where the horizontal axis represents the population proportion under the hypothetical data-generating process and each line indicates a different sample size. We observe that the power of a statistical test increases as the sample size becomes greater and the true population proportion p^* shifts away from the null value $p = 0.5$.

The above specific example illustrates the main principle of power analysis. We summarize the general procedure below.

Power is defined as the probability of rejecting the null hypothesis when the null hypothesis is false, which is equal to one minus the probability of type II error. **Power analysis** consists of the following steps:

1. Select the settings of the statistical hypothesis test you plan to use. This includes the specification of the test statistic, null and alternative hypotheses, and significance level.
2. Choose the population parameter value under a hypothetical data-generating process.
3. Compute the probability of rejecting the null hypothesis under this data-generating process with a given sample size.

One can then vary the sample size to examine how the power of the test changes to decide the **sample size** necessary for the desired level of power.

The power analysis can be conducted in a similar manner for two-sample tests. Consider the two-sample test of proportions, which can be used to analyze a randomized experiment with a binary outcome variable. The test statistic is the difference in sample proportion between the treatment and control groups, $\overline{Y}_{n_1} - \overline{X}_{n_0}$. Under the null hypothesis that this difference in the population, or the population average treatment effect (PATE), is equal to zero, the sampling distribution of the test statistic is given by $\mathcal{N}(0, p(1-p)(1/n_1 + 1/n_0))$, where p is the overall population proportion (see equation (7.20)), which is equal to the weighted average of the proportions in the two groups, $p = (n_0 p_0 + n_1 p_1)/(n_0 + n_1)$. To compute the power of the statistical test in this case, we must specify the population proportion separately for the treatment and control groups, p_1^* and p_0^*, under a hypothetical data-generating process. Then, the sampling distribution of the test statistic under this data-generating process is given by $\mathcal{N}(p_1^* - p_0^*, p_1^*(1-p_1^*)/n_1 + p_0^*(1-p_0^*)/n_0)$. Using this information, we can compute the probability of rejecting the null.

As an example, consider the résumé experiment analyzed in section 2.1. Suppose that we plan to send out 500 résumés with black-sounding names and another 500 résumés with white-sounding names. Further, assume that we expect the callback rate to be around 5% for black names and 10% for white names.

```
## parameters
n1 <- 500
n0 <- 500
p1.star <- 0.05
p0.star <- 0.1
```

To compute the power of this statistical test, we first compute the overall callback rate as a weighted average of callback rates of the two groups, where the weights are their sample size. We then compute the standard error under the null hypothesis, i.e., standard error $= \sqrt{p(1-p)(1/n_0 + 1/n_1)}$, as well as under the hypothetical data-generating process, i.e., standard error* $= \sqrt{p_1^*(1-p_1^*)/n_1 + p_0^*(1-p_0^*)/n_0}$.

```
## overall callback rate as a weighted average
p <- (n1 * p1.star + n0 * p0.star) / (n1 + n0)
## standard error under the null
se <- sqrt(p * (1 - p) * (1 / n1 + 1 / n0))
## standard error under the hypothetical data-generating process
se.star <- sqrt(p1.star * (1 - p1.star) / n1 + p0.star * (1 - p0.star) / n0)
```

We can now compute the power by calculating the probability that the difference in two proportions, $\overline{Y}_n - \overline{X}_n$, takes a value either less than $-z_{\alpha/2} \times$ standard error or greater than $-z_{\alpha/2} \times$ standard error*, under the hypothetical data-generating process.

```
pnorm(-cr.value * se, mean = p1.star - p0.star, sd = se.star) +
    pnorm(cr.value * se, mean = p1.star - p0.star,
          sd = se.star, lower.tail = FALSE)

## [1] 0.85228
```

While for illustration we computed the power by hand, we can use the `power.prop.test()` function available in R. This function, which is applicable to the two-sample test for proportions, can either compute the power given a set of parameters or determine a parameter value given a target power level. The arguments of this function include the sample size per group (`n`), population proportions for two groups (`p1.star` and `p2.star`), significance level (`sig.level`), and power (`power`). Note that the function assumes the two groups have an identical sample size, i.e., $n_0 = n_1$. To compute the power, we set `power = NULL` (default). The following syntax gives a result identical to what we computed above.

```
power.prop.test(n = 500, p1 = 0.05, p2 = 0.1, sig.level = 0.05)

##
##      Two-sample comparison of proportions power calculation
##
##               n = 500
##              p1 = 0.05
##              p2 = 0.1
##       sig.level = 0.05
##           power = 0.8522797
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

The `power.prop.test()` function also enables sample size calculation by simply setting the `power` argument to a desired level and setting `n` to `NULL` (default). For example, if we want to know, under the same conditions as above, the minimum sample size necessary to obtain a 90% level of power, we use the following R syntax. The result below implies that we need at least 582 observations per group in order to achieve this power.

```
power.prop.test(p1 = 0.05, p2 = 0.1, sig.level = 0.05, power = 0.9)
##
##      Two-sample comparison of proportions power calculation
##
##               n = 581.0821
##              p1 = 0.05
##              p2 = 0.1
##       sig.level = 0.05
##           power = 0.9
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

For continuous variables, we can conduct a power analysis based on *Student's t-test*, introduced in section 7.2.4. The logic is exactly the same as that described above for one-sample and two-sample tests of proportions. The `power.t.test()` function can perform a power analysis where the `type` argument specifies a two-sample (`"two.sample"`) or one-sample (`"one.sample"`) test. For a one-sample *t*-test, we must specify the mean `delta` and standard deviation `sd` of a normal random variable under a hypothetical data-generating process. For a two-sample *t*-test, the function assumes that the standard deviation and sample size are identical for the two groups. We, therefore, specify the true difference-in-means `delta` under a hypothetical data-generating process as well as a standard deviation `sd`. Finally, the function assumes the null hypothesis that the mean is zero for a one-sample test and the mean difference is zero for a two-sample test. If the null value is not zero, then one simply has to adjust the hypothetical data-generating process by subtracting that value from the true mean (or mean difference).

Below, we present two examples of using the `power.t.test()` function. The first is the power calculation for a one-sample test with a true mean of 0.25 and standard deviation of 1. The sample size is 100. Recall that the assumed mean value under the null hypothesis is zero.

```
power.t.test(n = 100, delta = 0.25, sd = 1, type = "one.sample")
##
##      One-sample t test power calculation
##
```

```
##                 n = 100
##             delta = 0.25
##                sd = 1
##         sig.level = 0.05
##             power = 0.6969757
##       alternative = two.sided
```

Under this setting, the power is calculated to be 70%. What is the sample size we need to have a power of 0.9 under the same setting? We can answer this question by specifying the `power` argument in the `power.t.test()` function while leaving the `n` argument unspecified.

```
power.t.test(power = 0.9, delta = 0.25, sd = 1, type = "one.sample")

##
##      One-sample t test power calculation
##
##                 n = 170.0511
##             delta = 0.25
##                sd = 1
##         sig.level = 0.05
##             power = 0.9
##       alternative = two.sided
```

The minimum sample size for obtaining a power of 0.9 or greater is 171. The second example is the sample size calculation for a one-sided two-sample test with a true mean difference of 0.25 and standard deviation of 1. We set the desired power to be 90%.

```
power.t.test(delta = 0.25, sd = 1, type = "two.sample",
             alternative = "one.sided", power = 0.9)

##
##      Two-sample t test power calculation
##
##                 n = 274.7222
##             delta = 0.25
##                sd = 1
##         sig.level = 0.05
##             power = 0.9
##       alternative = one.sided
##
## NOTE: n is number in *each* group
```

The result shows that we need a minimum of 275 observations per group to achieve a power of 90% under this setting.

7.3 Linear Regression Model with Uncertainty

As the final topic of this book, we consider the uncertainty of estimates based on the linear regression model introduced in chapter 4. In that chapter, we used the linear regression model mainly as a tool to make predictions. We also showed that when applied to a randomized controlled trial with binary treatments, the linear regression model can yield unbiased estimates of average treatment effects. In this section, we introduce another perspective that portrays the linear regression model as an approximation of the *data-generating process* in the real world. Under this framework, we can quantify the uncertainty of our estimates over repeated hypothetical sampling from the specified generative model. Once we view the linear regression model as a generative model, we can compute the standard errors and confidence intervals for our quantities of interest and conduct hypothesis testing.

7.3.1 LINEAR REGRESSION AS A GENERATIVE MODEL

Recall that the linear regression model with p predictors (explanatory or independent variables) is defined as

$$Y_i = \alpha + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{ip} + \epsilon_i. \tag{7.21}$$

In this model, Y represents the outcome or response variable, X_{ij} is the jth predictor, for $j = 1, 2, \ldots, p$, and ϵ_i denotes the unobserved error term for the ith observation. The model also has a total of $(p + 1)$ coefficients to be estimated, where α represents an intercept and β_j denotes a coefficient for the jth explanatory variable for $j = 1, 2, \ldots, p$.

According to this model, the outcome variable is generated as a linear function of the explanatory variables and the error term. For example, in section 4.2, we modeled the relationship between facial impressions and election outcomes using linear regression. In that application, the election outcome was a linear function of facial impressions and the error term. The error term contains all determinants of election outcomes that we do not observe, such as campaign resources, name recognition, and voter mobilization efforts.

In the model, the only variable we do not directly observe is the error term. As such, the key assumption of the model concerns the distribution of this random variable ϵ_i. Specifically, the linear regression model is based on the following *exogeneity* assumption.

The **exogeneity** assumption for the **linear regression model** is defined as

$$\mathbb{E}(\epsilon_i \mid \mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_p) = \mathbb{E}(\epsilon_i) = 0. \tag{7.22}$$

The assumption implies that the unobserved determinants of outcome, contained in the error term ϵ_i, are unrelated to all the observed predictors X_{ij} for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, p$. In this equation, $\mathbf{X}_j$ is an $n \times 1$ vector containing the jth covariate of all observations.

The assumption says that the *conditional expectation* of the error term given the explanatory variables, which is the first term in equation (7.22), is equal to its marginal or unconditional expectation, which is the second term in the equation and is equal to zero. The marginal expectation of the error term can always be assumed to be zero in the linear regression model so long as an intercept α is included in the model. The exogeneity assumption implies that the mean of the error term does not depend on the predictors or explanatory variables included in the model. In other words, the unobserved determinants of the outcome variable, which are contained in the error term, should be *uncorrelated* with all the observed predictors. In the election example, this implies that other, unobserved determinants of election outcomes should not be correlated with candidates' facial impressions.

In general, the conditional expectation of a random variable Y given another random variable X, denoted by $\mathbb{E}(Y \mid X)$, is the expectation of Y given a particular value of X. As such, this conditional expectation is a function of X, i.e., $\mathbb{E}(Y \mid X) = g(X)$, where $g(X)$ is called the *conditional expectation function*. All the definitions and rules of expectation introduced in section 6.3.5 hold for conditional expectation, except that we treat the variables in the conditioning set as fixed and compute the expectation with respect to the conditional distribution of Y given X. Thus, under the exogeneity assumption, the linear regression model assumes that the conditional expectation function for the outcome variable given the set of predictors is linear:

$$\mathbb{E}(Y_i \mid \boldsymbol{X}_1, \ldots, \boldsymbol{X}_p) = \alpha + \beta_1 X_{i1} + \cdots + \beta_p X_{ip}.$$

When deriving this result, we used the exogeneity assumption as well as the fact that the conditional expectation of $\beta_j X_{ij}$ given $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_p$ equals itself.

The **conditional expectation** of a random variable Y given another random variable X is denoted by $\mathbb{E}(Y \mid X)$ and is defined as

$$\mathbb{E}(Y \mid X) = \begin{cases} \sum_y y \times f(y \mid X) & \text{if } Y \text{ is discrete,} \\ \int y \times f(y \mid X) dy & \text{if } Y \text{ is continuous,} \end{cases}$$

where $f(Y \mid X)$ is the conditional probability mass function (conditional probability density function) of the discrete (continuous) random variable Y given X.

In *randomized controlled trials*, a violation of exogeneity does not occur because treatment assignment is randomized. In the framework of the linear regression model, this means that the treatment variable, which is represented by X, is statistically independent of all observed and unobserved pretreatment characteristics, which are contained in ϵ. Therefore, the exogeneity assumption is automatically satisfied. Consider the randomized controlled trial about women as policy makers described in section 4.3.1. In this experiment, the explanatory variable of interest, X, is whether seats in the local government, Gram Panchayat (GP), are reserved for female leaders. This variable is randomized and hence statistically independent of all other possible determinants of policy outcomes. For example, the number of new or repaired drinking water facilities in the village is likely to be determined not only by the existence of female leaders but also by numerous other factors such as the population size and the income level. Fortunately, we do not have to worry about these potential *unobserved confounders* because the randomized treatment assignment makes the treatment variable independent of these factors.

In *observational studies*, however, the exogeneity assumption may be violated. Suppose that the reservation of some GPs for female leaders is not randomized. Then, it is possible that villages with high levels of education and liberal ideologies are likely to elect female leaders for their GPs. Under this scenario, we cannot simply attribute the difference in the number of new or repaired drinking water facilities between villages to the gender of their politicians alone. It may be that highly educated villagers want better drinking water facilities and politicians are simply responding to the demands of their constituency. That is, both female and male politicians are responding to their constituencies, but their policy outcomes are different because they have different constituencies rather than because their genders are different. In observational studies, the unobserved confounders may be contained in the error term (e.g., education level of villagers), and if they are correlated with the observed explanatory variables (e.g., gender of politicians), the exogeneity assumption will be violated.

How can we address this problem of unobserved confounding in observational studies? In chapter 2, we learned that one strategy is to compare the treated units with similar control units. Ideally, we would like to find units that did not receive treatment and yet are similar to the treated units in terms of many observed characteristics. In the study on the minimum wage and employment described in section 2.5, researchers chose fast-food restaurants in Pennsylvania (PA), in which the minimum wage was not increased, as the control group for the fast-food restaurants in New Jersey (NJ), for which the minimum wage was raised. The idea was that since these restaurants are quite similar in their patterns of employment, products, and sales, we can use the restaurants in PA to infer the employment level of the restaurants in NJ that would have resulted if the minimum wage had not been increased. If there exist no unobserved factors, other than the treatment in NJ, that influence employment in NJ fast-food restaurants (i.e., no unobserved confounders), then the average difference in employment between the restaurants in NJ and those in PA can be attributed to the increase in NJ's minimum wage. The assumption of no unobserved confounding factors has several different names, including *unconfoundedness*, *selection on observables*, and *no omitted variables*, but they all mean the same thing.

The assumption of no unobserved confounding factors studied in chapter 2, therefore, is directly related to the exogeneity assumption under the linear regression model. Indeed, the exogeneity assumption will be violated whenever unobserved confounding variables exist. In the linear regression model framework, we can address this problem by measuring these confounders and including them as additional predictors in the model in order to adjust for their differences between the treatment and control groups. Although this strategy assumes a linear relationship between the outcome and these confounding variables, conceptually it is the same as comparing treated and control units that have similar characteristics. It can be shown that so long as all confounding variables are included in the model (and the linear relationship between the outcome and all explanatory variables holds), the estimated coefficient for the treatment variable represents an unbiased estimate of the average treatment effect.

In the minimum-wage example, assume that the only confounding factors between the fast-food restaurants in NJ and those in PA are the fast-food chain to which each restaurant belongs, its wage, and the proportion of full-time employment before the minimum wage was increased in NJ. Thus, we adjust for these three variables in the linear model, where the outcome variable is the proportion of full-time employment after the minimum wage was increased in NJ and the treatment variable is whether a restaurant is located in NJ. We use the data set described in table 2.5 and regress the outcome variable and three confounding variables using the `lm()` function. Before we fit the linear regression, we compute the proportion of full-time employment before and after the minimum wage was increased in NJ. We also create an indicator, or "dummy" variable, that equals 1 if a restaurant is located in NJ and 0 if it is in PA.

```
minwage <- read.csv("minwage.csv")
## compute proportion of full-time employment before minimum wage increase
minwage$fullPropBefore <- minwage$fullBefore /
    (minwage$fullBefore + minwage$partBefore)
## same thing after minimum-wage increase
minwage$fullPropAfter <- minwage$fullAfter /
    (minwage$fullAfter + minwage$partAfter)
## an indicator for NJ: 1 if it's located in NJ and 0 if in PA
minwage$NJ <- ifelse(minwage$location == "PA", 0, 1)
```

We now regress the proportion of full-time employment after the minimum-wage increase on the treatment variable (i.e., whether a restaurant is located in NJ) as well as on 3 other potential confounding variables. We note that `chain` is a factor variable with 4 different chains of fast-food restaurants. When a factor variable is used in the `lm()` function, as we saw in section 4.3.2, the function will automatically create the appropriate number of indicator variables for each category. In this case, since we have an intercept and the factor has 4 categories, the function will create 3 indicator variables. The `lm()` function by default includes an intercept. If we remove the intercept using the `-1` syntax, then it will create 1 indicator variable for each of the four categories. As explained in section 4.3.2, these two models are equivalent and yield an identical predicted value given the same values of the explanatory variables while yielding different estimates of coefficients.

```
fit.minwage <- lm(fullPropAfter ~ -1 + NJ + fullPropBefore +
                       wageBefore + chain, data = minwage)
## regression result
fit.minwage
##
## Call:
## lm(formula = fullPropAfter ~ -1 + NJ + fullPropBefore + wageBefore +
##     chain, data = minwage)
##
## Coefficients:
##              NJ  fullPropBefore      wageBefore
##         0.05422         0.16879         0.08133
## chainburgerking        chainkfc       chainroys
##        -0.11563        -0.15080        -0.20639
##     chainwendys
##        -0.22013
```

The result shows that the minimum-wage increase in NJ raised the proportion of full-time employees by 5.4 percentage points (represented by the estimated coefficient for the `NJ` variable) after adjusting for the proportion of full-time employees and wages before the minimum-wage increase as well as the chains of fast-food restaurants. By excluding the intercept, we can immediately compare the estimated coefficients across fast-food restaurant chains. We find that Burger King is predicted to have the highest proportion of full-time employment after adjusting for the other factors in the model. If we include an intercept, the estimated coefficients need to be interpreted relative to the base category, which will be dropped from the regression model. The base category of a factor variable represents a category to which the other categories of the variable are compared.

```
fit.minwage1 <- lm(fullPropAfter ~ NJ + fullPropBefore +
                       wageBefore + chain, data = minwage)
fit.minwage1
##
## Call:
## lm(formula = fullPropAfter ~ NJ + fullPropBefore + wageBefore +
##     chain, data = minwage)
##
## Coefficients:
##    (Intercept)              NJ  fullPropBefore
##       -0.11563         0.05422         0.16879
##     wageBefore        chainkfc       chainroys
##        0.08133        -0.03517        -0.09076
##    chainwendys
##       -0.10451
```

The `lm()` function excluded the indicator variable for Burger King from the regression, which means that the estimated coefficients for all other fast-food restaurant chains are relative to Burger King. Consistent with the previous result, we find that all other estimated coefficients are negative, indicating that Burger King is predicted to have the highest proportion of full-time employment after adjusting for the other factors in the model. We emphasize that these two models are equivalent, yielding the same predicted values. For example, we use the outputs of the two regression models to predict the outcome for the first observation in the data yielding an identical predicted value.

```
predict(fit.minwage, newdata = minwage[1, ])

##         1
## 0.2709367

predict(fit.minwage1, newdata = minwage[1, ])

##         1
## 0.2709367
```

Valid inference under the linear model assumes the **exogeneity assumption** given in equation (7.22). This assumption will be violated if there exist **unobserved confounders**. To make the exogeneity assumption more plausible, researchers can measure confounding variables and include them as additional explanatory variables in the linear regression model.

7.3.2 UNBIASEDNESS OF ESTIMATED COEFFICIENTS

How accurately can we estimate the coefficients of the linear regression model? Under the assumption that the linear regression model actually describes the true data-generating process, we consider the question of how to quantify the uncertainty associated with estimated coefficients. For simplicity, let us consider the model with one predictor only, though the results presented in this section can be generalized to linear regression with more than one predictor:

$$Y_i = \alpha + \beta X_i + \epsilon_i. \tag{7.23}$$

Recall from the discussion given in section 4.2.3 that if the linear regression model contains only an intercept and one predictor, then the least squares estimates are given by

$$\hat{\alpha} = \overline{Y} - \hat{\beta}\overline{X}, \tag{7.24}$$

$$\hat{\beta} = \frac{\sum_{i=1}^{n}(Y_i - \overline{Y})(X_i - \overline{X})}{\sum_{i=1}^{n}(X_i - \overline{X})^2}. \tag{7.25}$$

In this equation, $\overline{X}$ and $\overline{Y}$ represent the sample average of the predictor X_i and the outcome variable Y_i, respectively.

It turns out that under the exogeneity assumption these least squares coefficients, $\hat{\alpha}$ and $\hat{\beta}$, are unbiased for their corresponding true values, α and β, respectively. Formally, we may write $\mathbb{E}(\hat{\alpha}) = \alpha$ and $\mathbb{E}(\hat{\beta}) = \beta$. This means that if we generate the data according to this linear model, the least squares estimates of the coefficients will equal their true values, on average, across the hypothetically repeated data sets. Thus, the method of least squares produces unbiased estimates while minimizing the sum of squared residuals.

For those who are mathematically inclined, we show this important result analytically. Since we assume that the linear regression model is the true data-generating process, we substitute the linear model expression given in equation (7.23) into equation (7.24). Noting that the average outcome is given by $\overline{Y} = \alpha + \beta\overline{X} + \bar{\epsilon}$, we obtain the following expression for the estimated intercept:

$$\hat{\alpha} = \alpha + \beta\overline{X} + \bar{\epsilon} - \hat{\beta}\overline{X} = \alpha + (\beta - \hat{\beta})\overline{X} + \bar{\epsilon}.$$

This equation shows that the estimation error $\hat{\alpha} - \alpha$ is given by $(\beta - \hat{\beta})\overline{X} + \bar{\epsilon}$. Similarly, we use equation (7.23) to rewrite the estimated slope coefficient given in equation (7.25) as the sum of the true value β and the estimation error $\hat{\beta} - \beta$:

$$\hat{\beta} = \frac{\sum_{i=1}^{n}(\beta X_i + \epsilon_i - \beta\overline{X} - \bar{\epsilon})(X_i - \overline{X})}{\sum_{i=1}^{n}(X_i - \overline{X})^2} = \beta + \underbrace{\frac{\sum_{i=1}^{n}(\epsilon_i - \bar{\epsilon})(X_i - \overline{X})}{\sum_{i=1}^{n}(X_i - \overline{X})^2}}_{\text{estimation error}},$$

where we used the fact that $\sum_{i=1}^{n}\beta X_i = \sum_{i=1}^{n}\beta\overline{X}$.

We can further simplify the numerator of this estimation error, i.e., the second term in this equation:

$$\begin{aligned}
\sum_{i=1}^{n}(\epsilon_i - \bar{\epsilon})(X_i - \overline{X}) &= \sum_{i=1}^{n}\epsilon_i(X_i - \overline{X}) - \sum_{i=1}^{n}\bar{\epsilon}(X_i - \overline{X}) \\
&= \sum_{i=1}^{n}\epsilon_i(X_i - \overline{X}) - \bar{\epsilon}\underbrace{\left(\sum_{i=1}^{n}X_i - n\overline{X}\right)}_{=0} \\
&= \sum_{i=1}^{n}\epsilon_i(X_i - \overline{X}).
\end{aligned}$$

Therefore, we obtain the following final expression for the estimation error of the slope coefficient:

$$\hat{\beta} - \beta = \frac{\sum_{i=1}^{n}\epsilon_i(X_i - \overline{X})}{\sum_{i=1}^{n}(X_i - \overline{X})^2}. \tag{7.26}$$

As discussed in section 7.1.1, to prove the unbiasedness of $\hat{\beta}$, we must show that on average $\hat{\beta}$ equals its true value β over repeated (hypothetical) data-generating processes. Mathematically, we compute the expectation of $\hat{\beta}$ and show it is equal to β, i.e., $\mathbb{E}(\hat{\beta}) = \beta$. In this case, we first compute the conditional expectation of $\hat{\beta}$ given the explanatory variable vector X under the exogeneity assumption given in equation (7.22), then show $\mathbb{E}(\hat{\beta} \mid \mathbf{X}) = \beta$. This means that for a given value of X, we consider the hypothetical process of repeatedly generating the outcome variable Y by sampling the error term ϵ independent of X and then compute the least squares estimates $\hat{\alpha}$ and $\hat{\beta}$. While these estimates differ each time, on average they should equal the true values α and β, respectively.

We first calculate the conditional expectation of the estimated slope coefficient. Since the expectation is computed given the predictor vector $\boldsymbol{X}$, the only random variable is the error term ϵ. This means that the other terms can be considered as constants and taken out of the expectation:

$$\mathbb{E}(\hat{\beta} - \beta \mid \boldsymbol{X}) = \frac{1}{\sum_{i=1}^{n}(X_i - \overline{X})^2} \sum_{i=1}^{n} \mathbb{E}(\epsilon_i \mid \boldsymbol{X})(X_i - \overline{X}) = 0.$$

The second equality is implied by the exogeneity assumption $\mathbb{E}(\epsilon \mid \boldsymbol{X}) = 0$. Therefore, the estimated slope coefficient for X_i is unbiased conditional on the predictor. Using this result, we can also show that the estimated intercept is unbiased conditional on the predictor vector $\boldsymbol{X}$:

$$\mathbb{E}(\hat{\alpha} - \alpha \mid \boldsymbol{X}) = \mathbb{E}(\hat{\beta} - \beta \mid \boldsymbol{X})\overline{X} + \mathbb{E}(\bar{\epsilon} \mid \boldsymbol{X}) = 0.$$

The result follows from the fact that $\mathbb{E}(\hat{\beta} - \beta \mid \boldsymbol{X}) = 0$ (unbiasedness of $\hat{\beta}$) and $\mathbb{E}(\bar{\epsilon} \mid \boldsymbol{X}) = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}(\epsilon_i \mid \boldsymbol{X}) = 0$ (exogeneity). Since this means that given *any* value of the predictor vector $\boldsymbol{X}$ the estimated coefficients, $\hat{\alpha}$ and $\hat{\beta}$, are unbiased, conditional unbiasedness implies unbiasedness without conditioning, i.e., $\mathbb{E}(\hat{\alpha}) = \alpha$ and $\mathbb{E}(\hat{\beta}) = \beta$.

Under the exogeneity assumption, the least squares estimates of the coefficients in the linear regression model are unbiased.

The argument we just made, that conditional unbiasedness of estimated coefficients implies (unconditional) unbiasedness, can be made more generally and is called the *law of iterated expectation.*

The **law of iterated expectation** states that for any two random variables X and Y, the following equality holds:

$$\mathbb{E}(Y) = \mathbb{E}\{\mathbb{E}(Y \mid X)\}.$$

The inner expectation averages over Y given X, yielding a function of X, and the outer expectation averages this resulting conditional expectation function over X.

For example, let Y be an individual income and X be the racial group the individual belongs to. Then, in order to obtain the average income in a population, we could simply compute the mean of everyone's income $\mathbb{E}(Y)$ or first compute the average income for each racial category $\mathbb{E}(Y \mid X) = g(X)$ and then obtain the overall mean income by calculating the weighted average of race-specific means, where the weight is proportional to the size of racial group $\mathbb{E}(g(X))$. Applying the law of iterated expectation, we would formally conclude that the estimated coefficients are unbiased:

$$\begin{aligned}
\mathbb{E}(\hat{\alpha}) &= \mathbb{E}\{\mathbb{E}(\hat{\alpha} \mid \boldsymbol{X})\} = \mathbb{E}(\alpha) = \alpha, \\
\mathbb{E}(\hat{\beta}) &= \mathbb{E}\{\mathbb{E}(\hat{\beta} \mid \boldsymbol{X})\} = \mathbb{E}(\beta) = \beta.
\end{aligned}$$

7.3.3 STANDARD ERRORS OF ESTIMATED COEFFICIENTS

Now that we have established the unbiasedness of estimated coefficients, we consider their standard errors. The standard error of each estimated coefficient represents the (estimated) standard deviation of its sampling distribution (see section 7.1.2). The sampling distribution is produced through a hypothetically repeated sampling process, yielding different estimated coefficients across samples. The standard error quantifies the average variability of the estimated coefficient over this repeated sampling procedure.

As in the case of unbiasedness, we consider the linear regression model with one predictor for the sake of simplicity. We derive the variance of the sampling distribution of the estimated slope coefficient $\hat{\beta}$ and then take its square root to obtain the standard error. As in the case of bias, we first compute the conditional variance given the predictor X. Recall the discussion in section 6.3.5 that the variance of a random variable does not change even if we add a constant to it. Thus, the variance of the estimated coefficient $\hat{\beta}$ equals that of the estimation error $\hat{\beta} - \beta$ since β is an (albeit unknown) constant. Using equation (7.26), we obtain

$$\begin{aligned}\mathbb{V}(\hat{\beta} \mid \boldsymbol{X}) &= \mathbb{V}(\hat{\beta} - \beta \mid \boldsymbol{X}) \\ &= \mathbb{V}\left(\frac{\sum_{i=1}^{n} \epsilon_i (X_i - \overline{X})}{\sum_{i=1}^{n} (X_i - \overline{X})^2} \,\middle|\, \boldsymbol{X}\right) \\ &= \frac{1}{\{\sum_{i=1}^{n} (X_i - \overline{X})^2\}^2} \mathbb{V}\left(\sum_{i=1}^{n} \epsilon_i (X_i - \overline{X}) \,\middle|\, \boldsymbol{X}\right). \qquad (7.27)\end{aligned}$$

The third equality follows from equation (6.38) and the fact that the denominator is a function only of the predictor X, which is treated as a constant when computing the conditional variance given $\boldsymbol{X}$.

To further simplify the expression in equation (7.27), we assume *homoskedasticity* of the error term. That is, we assume that, conditional on the predictor X, the error term of observation i is independent of that of another observation, and that the variance of the error term does not depend on the predictor X.

The assumption of **homoskedastic error** consists of the following two components:

1. ϵ_i is independent of ϵ_j conditional on $\boldsymbol{X}$ for all $i \neq j$.
2. The variance of error does not depend on the predictor: $\mathbb{V}(\epsilon_i \mid \boldsymbol{X}) = \mathbb{V}(\epsilon_i)$.

Under this homoskedasticity assumption, we can further simplify the numerator of equation (7.27):

$$\mathbb{V}\left(\sum_{i=1}^{n} \epsilon_i (X_i - \overline{X}) \,\middle|\, \boldsymbol{X}\right) = \sum_{i=1}^{n} \mathbb{V}(\epsilon_i \mid \boldsymbol{X})(X_i - \overline{X})^2 = \mathbb{V}(\epsilon_i) \sum_{i=1}^{n} (X_i - \overline{X})^2. \quad (7.28)$$

Putting this together with equation (7.27), we arrive at the following variance of the estimated slope coefficient $\hat{\beta}$ under the homoskedasticity and exogeneity assumptions:

$$\mathbb{V}(\hat{\beta} \mid \boldsymbol{X}) = \frac{\mathbb{V}(\epsilon_i)}{\sum_{i=1}^{n}(X_i - \overline{X})^2}. \quad (7.29)$$

Although the above expression represents the conditional variance of $\hat{\beta}$ given the predictor X, we can also compute the unconditional variance of $\hat{\beta}$. The former is based on the variability of $\hat{\beta}$ under the hypothetical scenario of repeated sampling of Y_i given X_i (or equivalently ϵ_i given X_i because Y_i is a function of X_i and ϵ_i) for each observation, where X_i is fixed throughout. In contrast, the latter represents the uncertainty of $\hat{\beta}$ under a somewhat more natural data-generating process where Y_i and X_i (or equivalently ϵ_i and X_i) are jointly sampled from the population for each hypothetical realization of the data. To derive the unconditional variance of $\hat{\beta}$, we use the following *law of total variance*.

The **law of total variance** states that for any two random variables X and Y the following equality holds:

$$\mathbb{V}(Y) = \mathbb{V}\{\mathbb{E}(Y \mid X)\} + \mathbb{E}\{\mathbb{V}(Y \mid X)\}.$$

The first term represents the variance of conditional expectation and the second term represents the expectation of conditional variance.

In words, this law implies that the unconditional variance of random variable Y is equal to the sum of the variance of the conditional expectation of Y given X and the expectation of the conditional variance of Y given X. Applying the law of total variance, we can show that the unconditional variance of $\hat{\beta}$ can be derived as

$$\begin{aligned}
\mathbb{V}(\hat{\beta}) &= \mathbb{V}(\mathbb{E}(\hat{\beta} \mid \boldsymbol{X})) + \mathbb{E}\{\mathbb{V}(\hat{\beta} \mid \boldsymbol{X})\} \\
&= \underbrace{\mathbb{V}(\beta)}_{=0} + \mathbb{E}\left(\frac{\mathbb{V}(\epsilon_i)}{\sum_{i=1}^{n}(x_i - \overline{X})^2}\right) \\
&= \mathbb{V}(\epsilon_i)\mathbb{E}\left[\frac{1}{\sum_{i=1}^{n}(X_i - \overline{X})^2}\right]. \quad (7.30)
\end{aligned}$$

In the above equation, $\mathbb{V}(\beta) = 0$ because β is a constant. This implies that the unconditional variance of $\hat{\beta}$ is equal to the expected value of the conditional variance of $\hat{\beta}$, i.e., $\mathbb{V}(\hat{\beta}) = \mathbb{E}\{\mathbb{V}(\hat{\beta} \mid \boldsymbol{X})\}$. Thus, a good estimate of the conditional variance is also a good estimate of the unconditional variance.

Given this result, under the assumption of homoskedastic error, we can compute the standard error of $\hat{\beta}$ as an estimate of the unconditional variance given in equation (7.30). We do this by first estimating $\mathbb{V}(\epsilon_i)$ using the sample variance of residuals $\hat{\epsilon}_i = Y_i - \hat{\alpha} - \hat{\beta} X_i$, and then taking the square root of it. That is, if we denote the estimated conditional variance by $\widehat{\mathbb{V}(\hat{\beta})}$, then the standard error of $\hat{\beta}$ is

$$\text{standard error of } \hat{\beta} = \sqrt{\widehat{\mathbb{V}(\hat{\beta})}} = \sqrt{\frac{\frac{1}{n}\sum_{i=1}^{n}\hat{\epsilon}_i^2}{\sum_{i=1}^{n}(X_i - \overline{X})^2}}. \tag{7.31}$$

When estimating $\mathbb{V}(\epsilon_i)$, we used the fact that the sample mean of residuals is always zero,[3] i.e., $\frac{1}{n}\sum_{i=1}^{n}(\hat{\epsilon}_i - \bar{\hat{\epsilon}})^2 = \frac{1}{n}\sum_{i=1}^{n}\hat{\epsilon}_i^2$.

Finally, the standard errors derived above are based on the assumption of homoskedastic errors. If this assumption is violated, then the calculation of standard errors needs to be adjusted. For example, in randomized controlled trials, the variance may differ for the treatment and control groups. In fact, when we computed the standard error for the difference-in-means estimator, we separately calculated the variance for each group (see equation (7.18)). If the error variance depends on the predictor, we say that error is *heteroskedastic*. Although beyond the scope of this book, there are various ways to compute the standard errors that account for heteroskedastic errors. They are called *heteroskedasticity-robust standard errors*.

7.3.4 INFERENCE ABOUT COEFFICIENTS

Given the standard error derived above, we can compute the confidence intervals following the procedure described in section 7.1.3. Specifically, using the *central limit theorem*, we can show that as the sample size increases, the sampling distribution of $\hat{\beta}$ approaches a normal distribution centered around the mean:

$$z\text{-score of } \hat{\beta} = \frac{\hat{\beta} - \beta}{\text{standard error of } \hat{\beta}} \overset{\text{approx.}}{\sim} \mathcal{N}(0, 1). \tag{7.32}$$

Therefore, we can use the critical values based on the standard normal distribution to construct the $(1 - \alpha) \times 100\%$ level confidence interval below:

$$\text{CI}(\alpha) = [\hat{\beta} - z_{\alpha/2} \times \text{standard error},\ \hat{\beta} + z_{\alpha/2} \times \text{standard error}]. \tag{7.33}$$

We can also conduct a hypothesis test for the slope coefficient. For example, we can test the null hypothesis that the slope coefficient is equal to a particular value β_0. Most often, researchers use zero as the true value under the null hypothesis and ask whether or not the true coefficient for the predictor is equal to zero, i.e., $\beta_0 = 0$. Under the general hypothesis-testing framework developed in section 7.2, our null hypothesis is $H_0 : \beta = \beta_0$. The test statistic is the z-score, i.e., $z^* = (\hat{\beta} - \beta_0)/\text{standard error}$, and the sampling distribution of this test statistic z^* under the null hypothesis is the standard normal distribution. Therefore, we can compute the p-value using the CDF of the standard normal distribution. For example, the two-sided p-value is given by $2 \times P(Z \leq z^*)$, where Z is a standard normal random variable.

[3] Since the expectation of the error term is also zero and hence does not need to be estimated, we divide the sum of squared residuals by n instead of $n - 1$ often used for the sample variance calculation (see the discussion in section 2.6.2).

Just as for the analysis of randomized experiments, researchers often use a more conservative confidence interval based on Student's t-distribution (see section 7.1.5). Technically, if we make an additional assumption that the error term is normally distributed with mean zero and homoskedastic variance, then the sampling distribution of z^*, which is called the *t-statistic* in this setting, is given by, without approximation, Student's t-distribution with $n-2$ degrees of freedom. This contrasts with the asymptotic approximation based on the standard normal distribution without assuming a particular distribution for the error term. The degrees of freedom are $n-2$ because two parameters, α and β, are estimated from the data. Since Student's t-distribution has thicker tails than the standard normal distribution, we will have a greater critical value and as a result, obtain a wider confidence interval and a greater p-value.

As the first example to illustrate the results described above, we revisit the randomized experiment from chapter 4, examining the effects of women as policy makers in India (see section 4.3.1). The data set we analyze is contained in `women.csv` and the variable names and descriptions are given in table 4.7. Recall that after loading the data set from this study as a data frame `women`, we regressed the number of drinking water facilities in a village, `water`, on a binary variable `reserved`, indicating whether each GP is reserved for women. Conveniently, in R, all of the necessary information can be obtained by applying the `summary()` function to the output from the `lm()` function, which fits a linear regression model.

```
women <- read.csv("women.csv")
fit.women <- lm(water ~ reserved, data = women)
summary(fit.women)

##
## Call:
## lm(formula = water ~ reserved, data = women)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -23.991 -14.738  -7.865   2.262 316.009
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   14.738      2.286   6.446 4.22e-10 ***
## reserved       9.252      3.948   2.344   0.0197 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 33.45 on 320 degrees of freedom
## Multiple R-squared:  0.01688,Adjusted R-squared:  0.0138
## F-statistic: 5.493 on 1 and 320 DF,  p-value: 0.0197
```

We find that the point estimate of the slope coefficient is 9.252 and its standard error is 3.948. This output uses a conservative confidence interval based on Student's t-distribution. The t-statistic for the estimated slope coefficient is, therefore, 2.344. If the null hypothesis is that the slope coefficient is zero, then the two-sided p-value can be computed using Student's t-distribution with 320 degrees of freedom because the sample size is 322. In the summary output, this p-value is shown to be 0.0197. Therefore, using the $\alpha = 0.05$ level of statistical significance, we reject the null hypothesis that the slope coefficient is zero. The asterisks in the summary output indicate the level of statistical significance. We can compute confidence intervals using the `confint()` function, where the default significance level is 0.05. The level of statistical significance can be changed with the `level` argument.

```
confint(fit.women) # 95% confidence intervals
##                  2.5 %   97.5 %
## (Intercept) 10.240240 19.23640
## reserved     1.485608 17.01924
```

The result suggests that having the GP reserved for women is estimated to increase the number of drinking water facilities by 9.25 facilities with a 95% confidence interval of [1.49, 17.02]. As expected, we observe that the 95% confidence interval does not contain zero.

While the mathematical derivation is beyond the scope of this book, we can also compute the standard error and confidence interval of the estimated coefficients in a more general setting with multiple predictors. The `summary()` function can be applied to the output of the `lm()` function even with multiple predictors. For example, we can summarize the results of the linear regression model fitted to the minimum-wage data earlier in section 7.3.1.

```
summary(fit.minwage)
##
## Call:
## lm(formula = fullPropAfter ~ -1 + NJ + fullPropBefore + wageBefore +
##   chain, data = minwage)
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.48617 -0.18135 -0.02809  0.15127  0.75091
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## NJ                 0.05422    0.03321   1.633  0.10343
## fullPropBefore     0.16879    0.05662   2.981  0.00307 **
## wageBefore         0.08133    0.03892   2.090  0.03737 *
## chainburgerking -0.11563    0.17888  -0.646  0.51844
## chainkfc         -0.15080    0.18310  -0.824  0.41074
## chainroys        -0.20639    0.18671  -1.105  0.26974
## chainwendys      -0.22013    0.18840  -1.168  0.24343
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2438 on 351 degrees of freedom
## Multiple R-squared:  0.6349,Adjusted R-squared:  0.6277
## F-statistic: 87.21 on 7 and 351 DF,  p-value: < 2.2e-16
```

The summary output contains the relevant information for each of the estimated coefficients. In this observational study, we are interested in the average effect of increasing the minimum wage in NJ, which corresponds to the coefficient of the `NJ` variable. Thus, the average effect of the minimum-wage increase on the proportion of full-time employees in NJ is estimated to be 5.4 percentage points with a standard error of 3.3 percentage points. According to the result, we fail to reject the null hypothesis that the average effect of the minimum-wage increase is zero. In other words, we cannot preclude the possibility that the nonzero point estimate we obtained may be due to the sampling error under the scenario that the minimum-wage increase did not, on average, change the proportion of full-time employment. The p-values in this case are based on Student's t-distribution with 351 degrees of freedom because we have a total of 358 observations and 7 parameters to be estimated. To obtain the 95% confidence interval for this estimate, we can use the `confint()` function as before.

```
## confidence interval just for the "NJ" variable
confint(fit.minwage)["NJ", ]
##        2.5 %      97.5 %
## -0.01109295  0.11953297
```

As expected, the confidence interval contains zero, consistent with the result of the hypothesis test. However, a large portion of the confidence interval contains positive values, providing evidence that the minimum-wage increase in NJ may not have decreased the proportion of full-time employment.

The above summary output presents various other statistics. They include the *residual standard error*, which is the sample standard deviation of residuals. Since there are $(p + 1)$ parameters to be estimated, the number of degrees of freedom equals $(n - p - 1)$ instead of the usual $(n - 1)$ used when computing average. The residual standard error represents the average magnitude of residuals under the fitted model. The output also includes R^2, or the *coefficient of determination*, which represents the proportion of explained variation in the outcome (see section 4.2.6). As explained in section 4.3.2, the *adjusted* R^2 includes the adjustment due to the number of degrees of freedom, penalizing models with large numbers of predictors.

7.3.5 INFERENCE ABOUT PREDICTIONS

As shown in chapter 4, one of the main advantages of regression modeling is its ability to predict outcomes of interest. In the case of linear regression models, once we estimate the coefficients, we can use the model to predict an outcome variable given the values of the predictors in the model. Below, we show how to compute the standard error and construct confidence intervals for a prediction based on the linear regression model.

For the sake of simplicity, consider the linear regression model with a single predictor, i.e., $Y_i = \alpha + \beta X_i + \epsilon_i$. We are interested in obtaining the standard error of the predicted value from this model when the predictor X takes a particular value x:

$$\widehat{Y} = \hat{\alpha} + \hat{\beta}x.$$

To derive the variance of the predicted value $\widehat{Y}$, we must recognize the fact that $\hat{\alpha}$ and $\hat{\beta}$ are possibly correlated with each other. When two random variables, X and Y, are correlated, the variance of their sum is not equal to the sum of their variances. Instead, the variance of their sum includes their *covariance*, defined as follows.

Let X and Y be random variables. Their **covariance** is defined as

$$\begin{aligned}\operatorname{Cov}(X, Y) &= \mathbb{E}\{(X - \mathbb{E}(X))(Y - \mathbb{E}(Y))\} \\ &= \mathbb{E}(XY) - \mathbb{E}(X)\mathbb{E}(Y).\end{aligned}$$

The **correlation,** a standardized version of covariance, is given by

$$\operatorname{Cor}(X, Y) = \frac{\operatorname{Cov}(X, Y)}{\sqrt{\mathbb{V}(X)\mathbb{V}(Y)}}.$$

Sample correlation, or the correlation of a sample, was introduced in chapter 3 (see section 3.6.2). If the two random variables are independent of each other, their covariance and correlation are zero. In addition, the general formula for the variance of the sum of two (possibly dependent) random variables is given as

$$\mathbb{V}(X + Y) = \mathbb{V}(X) + \mathbb{V}(Y) + 2 \operatorname{Cov}(X, Y).$$

More generally,

$$\mathbb{V}(aX + bY + c) = a^2\mathbb{V}(X) + b^2(Y) + 2ab \operatorname{Cov}(X, Y)$$

where a, b, c are constants.

Since $\hat{\alpha}$ and $\hat{\beta}$ may not be independent, using the general formula introduced above, we obtain the following variance of predicted value $\widehat{Y}$ when the predictor X equals a particular value x:

$$\mathbb{V}(\widehat{Y}) = \mathbb{V}(\hat{\alpha} + \hat{\beta}x) = \mathbb{V}(\hat{\alpha}) + \mathbb{V}(\hat{\beta})x^2 + 2x \operatorname{Cov}(\hat{\alpha}, \hat{\beta}).$$

We can compute the standard error by estimating each component of this variance and then taking the square root of the estimated variance of $\widehat{Y}$:

$$\text{standard error of } \widehat{Y} = \sqrt{\widehat{\mathbb{V}(\hat{\alpha})} + \widehat{\mathbb{V}(\hat{\beta})}x^2 + 2x\, \widehat{\operatorname{Cov}(\hat{\alpha}, \hat{\beta})}}.$$

Once the standard error is calculated, we can apply the *central limit theorem* to approximate the sampling distribution of the z-score for the predicted value $\widehat{Y}$ using the standard normal distribution:

$$z\text{-score of } \widehat{Y} = \frac{\widehat{Y} - (\alpha + \beta x)}{\text{standard error of } \widehat{Y}} \overset{\text{approx.}}{\sim} \mathcal{N}(0,\ 1). \tag{7.34}$$

From this result, we can obtain a confidence interval and conduct a hypothesis test for a selected level of statistical significance.

As an example of inference with prediction, we revisit the regression discontinuity design introduced in section 4.3.4. In that study, we estimated the average effect of winning an election on a candidate's wealth in the United Kingdom. Instead of comparing members of Parliament (MPs) who won an election with those who lost it, researchers focused on those who narrowly won or narrowly lost an election. The idea was that if winning an election has a large effect on one's wealth, we should expect a substantial gap in the average wealth at the winning threshold, i.e., the winning margin of zero. Two linear regression models were used to predict the average wealth at this threshold, one based on narrow winners and the other fitted to narrow losers. Here, we reproduce the regression analysis conducted in section 4.3.4 separately for the Labour and Tory Parties.

```r
## load the data and subset them into two parties
MPs <- read.csv("MPs.csv")
MPs.labour <- subset(MPs, subset = (party == "labour"))
MPs.tory <- subset(MPs, subset = (party == "tory"))
## two regressions for Labour: negative and positive margin
labour.fit1 <- lm(ln.net ~ margin,
                  data = MPs.labour[MPs.labour$margin < 0, ])
labour.fit2 <- lm(ln.net ~ margin,
                  data = MPs.labour[MPs.labour$margin > 0, ])
## two regressions for Tory: negative and positive margin
tory.fit1 <- lm(ln.net ~ margin, data = MPs.tory[MPs.tory$margin < 0, ])
tory.fit2 <- lm(ln.net ~ margin, data = MPs.tory[MPs.tory$margin > 0, ])
```

The average treatment effect of winning an election results from predicting the average wealth at the winning threshold, i.e., a winning margin of zero. The confidence interval on the predicted value from each regression can be obtained by setting the `interval` argument in the `predict()` function to `"confidence"` rather than to `"none"`, which is the default. Note that as in the `confint()` function, the level of statistical significance can be selected by setting the `level` argument to a desired value (the default is 0.95). We focus on the Tory Party here.

```r
## Tory Party: prediction at the threshold
tory.y0 <- predict(tory.fit1, interval = "confidence",
                   newdata = data.frame(margin = 0))
```

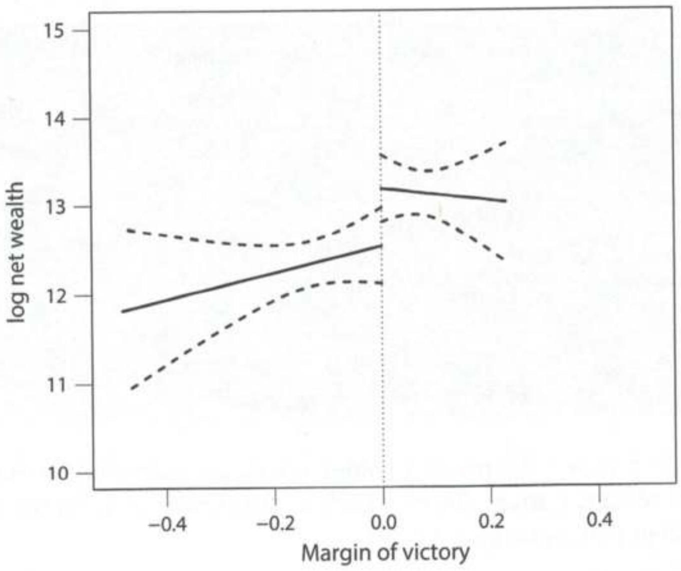
```
tory.y0
##        fit      lwr      upr
## 1 12.53812 12.11402 12.96221
tory.y1 <- predict(tory.fit2, interval = "confidence",
                    newdata = data.frame(margin = 0))
tory.y1
##       fit      lwr      upr
## 1 13.1878 12.80691 13.56869
```

In this output, the predicted value is given by `fit` and the lower and upper confidence bands are denoted by `lwr` and `upr`, respectively. For example, the average net wealth for non-MPs at the threshold is estimated to be 12.54 log net wealth with a 95% confidence interval of [12.11, 12.96]. Similarly, the average net wealth for MPs at the threshold is estimated to be 13.19 log net wealth with a 95% confidence interval of [12.81, 13.57]. The following code chunk plots these two regression lines (solid lines) with their 95% confidence intervals (dashed lines), using the range of predictor x. To do this, we first define the two ranges of the electoral margin and then compute the predictions for each range with 95% confidence intervals.

```
## range of predictors; min to 0 and 0 to max
y1.range <- seq(from = 0, to = min(MPs.tory$margin), by = -0.01)
y2.range <- seq(from = 0, to = max(MPs.tory$margin),by = 0.01)
## prediction using all the values
tory.y0 <- predict(tory.fit1, interval = "confidence",
                   newdata = data.frame(margin = y1.range))
tory.y1 <- predict(tory.fit2, interval = "confidence",
                   newdata = data.frame(margin = y2.range))
```

Finally, we plot the results where the solid lines represent the predicted values and the dashed lines represent the confidence intervals.

```
## plotting the first regression with losers
plot(y1.range, tory.y0[, "fit"], type = "l", xlim = c(-0.5, 0.5),
     ylim = c(10, 15), xlab = "Margin of victory", ylab = "log net wealth")
abline(v = 0, lty = "dotted")
lines(y1.range, tory.y0[, "lwr"], lty = "dashed") # lower CI
lines(y1.range, tory.y0[, "upr"], lty = "dashed") # upper CI
## plotting the second regression with winners
lines(y2.range, tory.y1[, "fit"], lty = "solid")  # point estimates
lines(y2.range, tory.y1[, "lwr"], lty = "dashed") # lower CI
lines(y2.range, tory.y1[, "upr"], lty = "dashed") # upper CI
```



In the plot, we observe that the width of the confidence interval widens as it moves away from the mean value of the predictor. While these two confidence intervals overlap with each other, what we really would like to do is to compute the confidence interval for the difference between these two predicted values. This is because the difference between the two predicted values represents the estimated average treatment effect at the threshold under the regression discontinuity design. Moreover, these two predicted values are assumed to be independent because they are based on two regression models that are fitted to two separate sets of observations. This means that the variance of the difference is the sum of the two variances. To compute the standard error of the difference in the predicted values, we obtain the standard error from each fitted regression. We then use the following formula to compute the standard error of the estimated difference:

$$\text{standard error of } (\widehat{Y}_1 - \widehat{Y}_0) = \sqrt{(\text{standard error of } \widehat{Y}_1)^2 + (\text{standard error of } \widehat{Y}_0)^2}.$$

In R, we obtain the standard error of a predicted value by setting the `se.fit` argument to `TRUE`. There are multiple elements in the output list of the `predict()` function when using this standard error option. Each element can be extracted from this list by using the symbol `$`.

```
## recompute the predicted value and return standard errors
tory.y0 <- predict(tory.fit1, interval = "confidence",  se.fit = TRUE,
                   newdata = data.frame(margin = 0))
tory.y0
## $fit
##        fit      lwr      upr
## 1 12.53812 12.11402 12.96221
```

```
##
## $se.fit
## [1] 0.2141793
##
## $df
## [1] 119
##
## $residual.scale
## [1] 1.434283
tory.y1 <- predict(tory.fit2, interval = "confidence", se.fit = TRUE,
                   newdata = data.frame(margin = 0))
```

Since in this case the predicted value equals the estimated intercept, the standard error one obtains through the `predict()` function is equal to the standard error of the intercept in the summary output.

```
## s.e. of the intercept is the same as s.e. of the predicted value
summary(tory.fit1)
##
## Call:
## lm(formula = ln.net ~ margin, data = MPs.tory[MPs.tory$margin <
##     0, ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.3195 -0.4721 -0.0349  0.6629  3.5798
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  12.5381     0.2142  58.540   <2e-16 ***
## margin        1.4911     1.2914   1.155    0.251
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.434 on 119 degrees of freedom
## Multiple R-squared: 0.01108, Adjusted R-squared: 0.002769
## F-statistic: 1.333 on 1 and 119 DF,  p-value: 0.2506
```

We can now compute the standard error of the estimated difference in the average log net wealth between MPs and non-MPs at the winning threshold. Using this standard error, we compute the confidence interval and conduct a hypothesis test for which the null hypothesis is that winning an election has zero average effect on candidates' net log wealth.

```
## standard error
se.diff <- sqrt(tory.y0$se.fit^2 + tory.y1$se.fit^2)
se.diff

## [1] 0.2876281

## point estimate
diff.est <- tory.y1$fit[1, "fit"] - tory.y0$fit[1, "fit"]
diff.est

## [1] 0.6496861

## confidence interval
CI <- c(diff.est - se.diff * qnorm(0.975), diff.est + se.diff * qnorm(0.975))
CI

## [1] 0.0859455 1.2134268

## hypothesis test
z.score <- diff.est / se.diff
p.value <- 2 * pnorm(abs(z.score), lower.tail = FALSE) # two-sided p-value
p.value

## [1] 0.02389759
```

We find that even though the confidence intervals of the two estimates overlap with each other, the difference between these two estimates is statistically significantly different from zero. Indeed, the average effect of winning an election is estimated to be 0.65 net log wealth with a 95% confidence interval of [0.09, 1.21], which does not contain zero. As a result, the two-sided p-value is less than the conventional statistical significance level, 0.05, allowing us to reject the null hypothesis of zero average effect. Thus, our analysis suggests that winning an election had a positive impact on candidates' net wealth. The overlap of the confidence intervals of the two estimates does not necessarily imply that the confidence interval of the difference between the two estimates contains zero.

7.4 Summary

In this chapter, we introduced a framework for methods of statistical inference that enables us to quantify the degree of uncertainty regarding our estimates. While we can never know how close our estimates are to the unknown truth, we can evaluate the performance of our estimators using a hypothetically repeated randomization of treatment assignment and/or repeated random sampling. We introduced the concept of **unbiasedness**. An unbiased estimator accurately estimates the parameter of interest on average over a hypothetically repeated data-generating process. Using the law of large numbers, we can also show that some estimators have the property of **consistency**, which implies that as the sample size increases, they converge to the true parameter values.

While unbiasedness is an attractive property, we also need to understand the precision of an estimator given that we can obtain only one realization of the estimator. We use **standard error** to quantify how far our estimator is from the true parameter value on average over a repeated data-generating process. Standard error is an estimate of the standard deviation of the sampling distribution of an estimator. Based on standard errors, we can also construct **confidence intervals**, which will contain the true parameter values with a prespecified probability, again over a repeated data-generating process. We also showed how to conduct a statistical **hypothesis test** by specifying a null hypothesis and examining whether or not the observed data are consistent with this hypothesis. We applied these inferential methods to the analysis of randomized experiments and sample surveys from earlier in this book.

Finally, we introduced **model-based inference**. We used a linear regression model as a probabilistic generative model, from which the data are assumed to be drawn. Under this setting, we can quantify the uncertainty of our estimated coefficients and predicted values. We showed that the least squares estimates of coefficients are unbiased and derived their standard errors. Using these results, we also explained how to construct confidence intervals and conduct hypothesis tests. Similarly, we showed how to quantify the uncertainty about our predicted values and applied the methodology to the regression discontinuity design introduced in an earlier chapter. These statistical methods play an essential role in our inference because they enable us to separate signals from noise, extracting systematic patterns from data.

7.5 Exercises

7.5.1 SEX RATIO AND THE PRICE OF AGRICULTURAL CROPS IN CHINA

In this exercise, we consider the effect of a change in the price of agricultural goods whose production and cultivation are dominated by either men or women.[4] Our data come from China, where centrally planned production targets during the Maoist era led to changes in the prices of major staple crops. We focus here on tea, the production and cultivation of which required a large female labor force, as well as orchard fruits, for which the labor force was overwhelmingly male. We use price increases brought on by government policy change in 1979 as a proxy for increases in sex-specific income, and ask the following question: Do changes in sex-specific income alter the incentives for Chinese families to have children of one gender over another? The CSV data file, `chinawomen.csv`, contains the variables shown in table 7.4, with each observation representing a particular Chinese county in a given year. Note that `post` is an indicator variable that takes the value `1` in a year following the policy change and `0` in a year before the policy change.

1. We begin by examining sex ratios in the postreform period (i.e., the period after 1979) according to whether or not tea crops were sown in the region.

[4] This exercise is based on Nancy Qian (2008) "Missing women and the price of tea in China: The effect of sex-specific earnings on sex imbalance." *Quarterly Journal of Economics*, vol. 123, no. 3, pp. 1251–1285.

Table 7.4. Chinese Births and Crops Data.

Variable	*Description*
`admin`	unique county identifier
`birpop`	birth population in a given year
`biryr`	year of cohort (birth year)
`cashcrop`	quantity of cash crops planted in the county
`orch`	quantity of orchard-type crops planted in the county
`teasown`	quantity of tea sown in the county
`sex`	proportion of males in the birth cohort
`post`	indicator variable for the introduction of price reforms

Estimate the mean sex ratio in 1985, which we define as the proportion of male births, separately for tea-producing and non-tea-producing regions. Compute the 95% confidence interval for each estimate by assuming independence across counties within a year. Note that we will maintain this assumption throughout this exercise. Furthermore, compute the difference-in-means between the two regions and its 95% confidence interval. Are sex ratios different across these regions? What assumption is required in order for us to interpret this difference as causal?

2. Repeat the analysis in the previous question for subsequent years, i.e., 1980, 1981, 1982, ..., 1990. Create a graph which plots the difference-in-means estimates and their 95% confidence intervals against years. Give a substantive interpretation of the plot.

3. Next, we compare tea-producing and orchard-producing regions before the policy enactment. Specifically, we examine the sex ratio and the proportion of Han Chinese in 1978. Estimate the mean difference, its standard error, and 95% confidence intervals for each of these measures between the two regions. What do the results imply about the interpretation of the results given in question 1?

4. Repeat the analysis for the sex ratio in the previous question for each year before the reform, i.e., from 1962 until 1978. Create a graph which plots the difference-in-means estimates between the two regions and their 95% confidence intervals against years. Give a substantive interpretation of the plot.

5. We will adopt the difference-in-differences design by comparing the sex ratio in 1978 (right before the reform) with that in 1980 (right after the reform). Focus on a subset of counties that do not have missing observations in these two years. Compute the difference-in-differences estimate and its 95% confidence interval. Note that we assume independence across counties but account for possible dependence across years within each county. Then, the variance of the

difference-in-differences estimate is given by

$$\begin{aligned}
&\mathbb{V}\{(\overline{Y}_{\text{tea,after}} - \overline{Y}_{\text{tea,before}}) - (\overline{Y}_{\text{orchard,after}} - \overline{Y}_{\text{orchard,before}})\} \\
&\quad = \mathbb{V}(\overline{Y}_{\text{tea,after}} - \overline{Y}_{\text{tea,before}}) + \mathbb{V}(\overline{Y}_{\text{orchard,after}} - \overline{Y}_{\text{orchard,before}}),
\end{aligned}$$

where dependence across years is given by

$$\begin{aligned}
&\mathbb{V}(\overline{Y}_{\text{tea,after}} - \overline{Y}_{\text{tea,before}}) \\
&\qquad = \mathbb{V}(\overline{Y}_{\text{tea,after}}) - 2\,\text{Cov}(\overline{Y}_{\text{tea,after}}, \overline{Y}_{\text{tea,before}}) + \mathbb{V}(\overline{Y}_{\text{tea,before}}) \\
&\qquad = \frac{1}{n}\left\{\mathbb{V}(Y_{\text{tea,after}}) - 2\,\text{Cov}(Y_{\text{tea,after}}, Y_{\text{tea,before}}) + \mathbb{V}(Y_{\text{tea,before}})\right\}.
\end{aligned}$$

A similar formula can be given for orchard-producing regions. What substantive assumptions does the difference-in-differences design require? Give a substantive interpretation of the results.

7.5.2 FILE DRAWER AND PUBLICATION BIAS IN ACADEMIC RESEARCH

The peer review process is the main mechanism through which scientific communities decide whether a research paper should be published in academic journals.[5] By having other scientists evaluate research findings, academic journals hope to maintain the quality of their published articles. However, some have warned that the peer review process may yield undesirable consequences. In particular, the process may result in *publication bias* wherein research papers with statistically significant results are more likely to be published. To make matters worse, by being aware of such a bias in the publication process, researchers may be more likely to report findings that are statistically significant and ignore others. This is called *file drawer bias.*

In this exercise, we will explore these potential problems using data on a subset of experimental studies that were funded by the Time-Sharing Experiments in the Social Sciences (TESS) program. This program is sponsored by the National Science Foundation (NSF). The data set necessary for this exercise can be found in the CSV files `filedrawer.csv` and `published.csv`. The `filedrawer.csv` file contains information about 221 research projects funded by the TESS program. However, not all of those projects produced a published article. The `published.csv` file contains information about 53 published journal articles based on TESS projects. This data set records the number of experimental conditions and outcomes and how many of them are actually reported in the published article. Tables 7.5 and 7.6 present the names and descriptions of the variables from these data sets.

1. We begin by analyzing the data contained in the `filedrawer.csv` file. Create a contingency table for the publication status of papers and the statistical

[5] This exercise is based on the following studies: Annie Franco, Neil Malhotra, and Gabor Simonovits (2014) "Publication bias in the social sciences: Unlocking the file drawer." *Science*, vol. 345, no. 6203, pp. 1502–1505 and Annie Franco, Neil Malhotra, and Gabor Simonovits (2015) "Underreporting in political science survey experiments: Comparing questionnaires to published results." *Political Analysis*, vol. 23, pp. 206–312.

Table 7.5. File Drawer and Publication Bias Data I.

Variable	*Description*
`id`	study identifier
`DV`	publication status
`IV`	statistical significance of the main findings
`max.h`	H-index (highest among authors)
`journal`	discipline of the journal for published articles

Table 7.6. File Drawer and Publication Bias Data II.

Variable	*Description*
`id.p`	paper identifier
`cond.s`	number of conditions in the study
`cond.p`	number of conditions presented in the paper
`out.s`	number of outcome variables in the study
`out.p`	number of outcome variables used in the paper

significance of their main findings. Do we observe any distinguishable trend towards the publication of strong results? Provide a substantive discussion.

2. We next examine whether there exists any difference in the publication rate of projects with strong versus weak results as well as with strong versus null results. To do so, first create a variable that takes the value of 1 if a paper was published and 0 if it was not published. Then, perform two-tailed tests of the difference in the publication rates for the aforementioned comparisons of groups, using 95% as the significance level. Briefly comment on your findings.

3. Using Monte Carlo simulations, derive the distribution of the test statistic under the null hypothesis of no difference for each of the two comparisons you made in the previous question. Do you attain similar p-values (for a two-tailed test) to those obtained in the previous question?

4. Conduct the following power analysis for a one-sided hypothesis test where the null hypothesis is that there is no difference in the publication rate between the studies with strong results and those with weak results. The alternative hypothesis is that the studies with strong results are less likely to be published than those with weak results. Use 95% as the significance level and assume that the publication rate for the studies with weak results is the same as the observed publication rate for those studies in the data. How many studies do we need in order to detect a 5 percentage point difference in the publication rate and for the test to attain

a power of 95%? For the number of observations in the data, what is the power of the test of differences in the publication rates?

5. The H-index is a measure of the productivity and citation impact of each researcher in terms of publications. More capable researchers may produce stronger results. To shed more light on this issue, conduct a one-sided test for the null hypothesis that the mean H-index is lower or equal for projects with strong results than those with null results. What about the comparison between strong versus weak results? Do your findings threaten those presented for question 2? Briefly explain.

6. Next, we examine the possibility of file drawer bias. To do so, we will use two scatter plots, one that plots the total number of conditions in a study (horizontal axis) against the total number of conditions included in the paper (vertical axis). Make the size of each dot proportional to the number of corresponding studies, via the `cex` argument. The second scatter plot will focus on the number of outcomes in the study (horizontal axis) and the number of outcomes presented in the published paper (vertical axis). As in the previous plot, make sure each circle is weighted by the number of cases in each category. Based on these plots, do you observe problems in terms of underreporting?

7. Create a variable that represents the total number of possible hypotheses to be tested in a paper by multiplying the total number of conditions and outcomes presented in the questionnaires. Suppose that these conditions yield no difference in the outcome. What is the average (per paper) probability that at the 95% significance level we reject at least one null hypothesis? What about the average (per paper) probability that we reject at least two or three null hypotheses? Briefly comment on the results.

7.5.3 THE 1932 GERMAN ELECTION IN THE WEIMAR REPUBLIC

Who voted for the Nazis? Researchers attempted to answer this question by analyzing aggregate election data from the 1932 German election during the Weimar Republic.[6] We analyze a simplified version of the election outcome data, which records, for each precinct, the number of eligible voters as well as the number of votes for the Nazi party. In addition, the data set contains the aggregate occupation statistics for each precinct. Table 7.7 presents the variable names and descriptions of the CSV data file `nazis.csv`. Each observation represents a German precinct.

The goal of the analysis is to investigate which types of voters (based on their occupation category) cast ballots for the Nazi party in 1932. One hypothesis says that the Nazis received much support from blue-collar workers. Since the data do not directly tell us how many blue-collar workers voted for the Nazis, we must infer this

[6] This exercise is based on the following article: G. King, O. Rosen, M. Tanner, A.F. Wagner (2008) "Ordinary economic voting behavior in the extraordinary election of Adolf Hitler." *Journal of Economic History*, vol. 68, pp. 951–996.

Table 7.7. 1932 German Election Data.

Variable	*Description*
`shareself`	proportion of self-employed potential voters
`shareblue`	proportion of blue-collar potential voters
`sharewhite`	proportion of white-collar potential voters
`sharedomestic`	proportion of domestically employed potential voters
`shareunemployed`	proportion of unemployed potential voters
`nvoter`	number of eligible voters
`nazivote`	number of votes for Nazis

information using a statistical analysis with certain assumptions. Such an analysis, where researchers try to infer individual behaviors from aggregate data, is called *ecological inference*.

To think about ecological inference more carefully in this context, consider the following simplified table for each precinct i.

	Occupation		
	Blue-collar	Non-blue-collar	
Vote choice			
Nazis	W_{i1}	W_{i2}	Y_i
Other parties or abstention	$1 - W_{i1}$	$1 - W_{i2}$	$1 - Y_i$
	X_i	$1 - X_i$	

The data at hand tells us only the proportion of blue-collar voters X_i and the vote share for the Nazis Y_i in each precinct, but we would like to know the Nazi vote share among the blue-collar voters W_{i1} and among the non-blue-collar voters W_{i2}. Then, there is a deterministic relationship between X, Y, and $\{W_1, W_2\}$. Indeed, for each precinct i, we can express the overall Nazi vote share as the weighted average of the Nazi vote share of each occupation:

$$Y_i = X_i W_{i1} + (1 - X_i) W_{i2}. \tag{7.35}$$

1. We exploit the linear relationship between the Nazi vote share Y_i and the proportion of blue-collar voters X_i given in equation (7.35) by regressing the former on the latter. That is, fit the following linear regression model:

$$\mathbb{E}(Y_i \mid X_i) = \alpha + \beta X_i. \tag{7.36}$$

   Compute the estimated slope coefficient, its standard error, and the 95% confidence interval. Give a substantive interpretation of each quantity.

2. Based on the fitted regression model from the previous question, predict the average Nazi vote share Y_i given various proportions of blue-collar voters X_i. Specifically, plot the predicted value of Y_i (the vertical axis) against various values

of X_i within its observed range (the horizontal axis) as a solid line. Add 95% confidence intervals as dashed lines. Give a substantive interpretation of the plot.

3. Fit the following alternative linear regression model:

$$\mathbb{E}(Y_i \mid X_i) = \alpha^* X_i + (1 - X_i)\beta^*. \tag{7.37}$$

   Note that this model does not have an intercept. How should one interpret α^* and β^*? How are these parameters related to the linear regression model given in equation (7.36)?

4. Fit a linear regression model where the overall Nazi vote share is regressed on the proportion of each occupation. The model should contain no intercept and 5 predictors, each representing the proportion of a certain occupation type. Interpret the estimate of each coefficient and its 95% confidence interval. What assumption is necessary to permit your interpretation?

5. Finally, we consider a model-free approach to ecological inference. That is, we ask how much we can learn from the data alone without making an additional modeling assumption. Given the relationship in equation (7.35), for each precinct, obtain the smallest value that is logically possible for W_{i1} by considering the scenario in which all non-blue-collar voters in precinct i vote for the Nazis. Express this value as a function of X_i and Y_i. Similarly, what is the largest possible value for W_{i1}? Calculate these bounds, keeping in mind that the value for W_{i1} cannot be negative or greater than 1. Finally, compute the bounds for the nationwide proportion of blue-collar voters who voted for the Nazis (i.e., combining the blue-collar voters from all precincts by computing their weighted average based on the number of blue-collar voters). Give a brief substantive interpretation of the results.

Chapter 8

Next

Statistics are no substitute for judgment.
— Henry Clay

What comes next? There are several directions one could take in order to further improve data analysis skills. The current book is a first course in applied data analysis and introduces only a tiny fraction of useful data analytic methods. There is much more to learn. An obvious next step is to learn more about data analysis and statistics. For example, one might enroll in a second course in data analysis and statistics (or read a relevant textbook) that covers regression modeling techniques, which are essential tools for quantitative social science. Another possibility is to take a course on specific topics of interest, such as causal inference, social network analysis, and survey methodology.

As an introduction to quantitative social science, this book does not take a mathematical approach to data analysis. Instead, the focus of the book is to give readers a sense of how data analysis is used in quantitative social science research, while teaching elementary concepts and methods. But since all of data analysis and statistical methods have a mathematical foundation, a deeper understanding of them requires a good command of mathematics. A better grasp of methods will, in turn, enable one to become a more sophisticated user of data analysis and statistics who can critically assess the advantages and limitations of various methodologies in applied research. Furthermore, if one is interested in becoming a methodologist who develops new methods, a solid foundation in mathematics is critical. In particular, it is essential to learn multivariate calculus and linear algebra, followed by probability theory. After these foundations, students can learn statistical theory and various modeling strategies in a rigorous fashion.

Since the main focus of this book is data analysis, we did not discuss how to collect data—yet without data collection, there would be no data analysis. Although we analyzed the data from several randomized controlled trials in this book, little attention was given to experimental designs. How should we recruit subjects when conducting an experiment? What are the experimental design strategies one could use in order to obtain precise estimates of causal effects? These and other questions arise when designing experiments in the laboratory and field. A pioneer statistician, Ronald A. Fisher, once stated, "To call in the statistician after the experiment is done may be no more than asking him to perform a postmortem examination: he may be able to say what the experiment died of."[1] We must learn how to design randomized experiments in order to take advantage of this powerful tool for causal inference. Even for observational studies, careful planning is required in order to identify the instances in which researchers can draw causal inference in a credible manner. Research design forms a fundamental component of quantitative social science research.

Similarly, while we analyze survey data in this book, we do not examine survey sampling strategy and questionnaire design. In many cases, the simple random sampling we discussed is not feasible, because we do not have a sampling frame that contains a complete list of all individuals of a target population. For example, when studying a population that is difficult to reach (e.g., homeless people, seasonal migrants), other strategies, such as respondent-driven sampling, have been used. Another important question is how to correct for the lack of representativeness in survey data. In particular, Internet surveys are now commonly used, but an online panel is often far from being representative of a target population. Questionnaire design also plays an essential role in obtaining accurate measurements. In chapter 3, we saw examples of a special technique for eliciting truthful answers to sensitive questions. The exercise in section 3.9.2 introduced a survey methodology that reduces measurement error due to the possibility that respondents may interpret the same questions differently. These examples suggest that studying a variety of data collection strategies is as important for quantitative social scientists as learning about data analysis.

While different interests may take people in various directions after completing this book, everyone should continue to practice data analysis. In the words of John W. Tukey,[2] "If data analysis is to be helpful and useful, it must be practiced." Now that users of this book have learned the basic methodology and programming necessary for data analysis, they should begin to conduct quantitative social science research by analyzing data sets of their choice. Just as with data analysis, one learns how to conduct research only by doing, not by reading the research of other people. With the massive amount of data available online, anyone from undergraduate to graduate students and from practitioners to academic researchers should be able to start making their own data-driven discoveries.

This book highlights the power of data analysis. However, it is also important to be aware of its fundamental limitations when analyzing data. In particular, data analysis is far from objective. Good data analysis must be accompanied by sound judgment, which is in turn built upon one's knowledge and experience. Without substantive theories, data analysis can easily be misguided. In quantitative social science research, we analyze data for the purpose of better understanding society and human behavior. This goal is unattainable unless we use social science theories to determine how data should be analyzed. Stronger theoretical guidance is required for the analysis of "big data" because without it we will not know where to look for interesting patterns.

[1] Ronald A. Fisher (1938) "Presidential address: The first session of the Indian Statistical Conference, Calcutta, 1938." *Sankhyā*, vol. 4, pp. 14–17.

[2] John W. Tukey (1962) "The future of data analysis." *Annals of Mathematical Statistics*, vol. 33, no. 1, pp. 1–67.

Although a solid grasp of the mathematics that underlie statistical theories and methods is important, we should not underestimate the value of contextual knowledge about the data sets to be analyzed. For example, to competently design and analyze the survey of Afghan civilians introduced in chapter 3, researchers had to understand the cultural, political, and economic environment of local communities in Afghanistan where the respondents live. Interviewing individuals who have little education, during a civil war, is a challenging task. The researchers worked with a local survey firm in order to gain access to rural villages through negotiation with local leaders and militants. For cultural reasons, they were unable to interview female respondents, and interviews had to take place in a public sphere where village elders were able to listen to survey questions and answers. Randomized response methodology is a classic survey method for asking sensitive questions while protecting the secrecy of individual responses. However, this method was seen as inappropriate in the study because the required randomization using coins or dice was considered to be against Islamic law. Other challenges in this study included how to ask respondents' tribal affiliation, how to measure the level of wealth when the economy is largely informal, and what policy questions to ask when measuring respondents' political ideology.

These examples illustrate the importance of contextual knowledge in designing and implementing quantitative social science research. Therefore, data analysts should learn about the relevant substance and background of their study, either on their own or by partnering with experts, well before starting to analyze data. They should also be aware of the danger that mechanical applications of statistical methods to data may lead to unreliable empirical findings. Indeed, this is the reason why applied statistics has developed separately in a variety of fields of the natural and social sciences. While statistical methods rest on universal mathematical theory and are widely applicable, their application requires specific substantive knowledge. The goal of this book has been to illustrate this unique feature of data analysis and statistics by showing how general methods can be used to answer interesting social science questions.

With rapid advancements in technology and data availability, the world needs those who can creatively combine substantive knowledge with data analysis skills in all fields, from academia to journalism. This book opens the door to this exciting world of data analysis.

General Index

R Index